マイコン別冊

パソコン移植テクニック

各機種
BASIC
比較研究!!

PART 1
入門編
BASIC
大研究



監修 丹治佐一

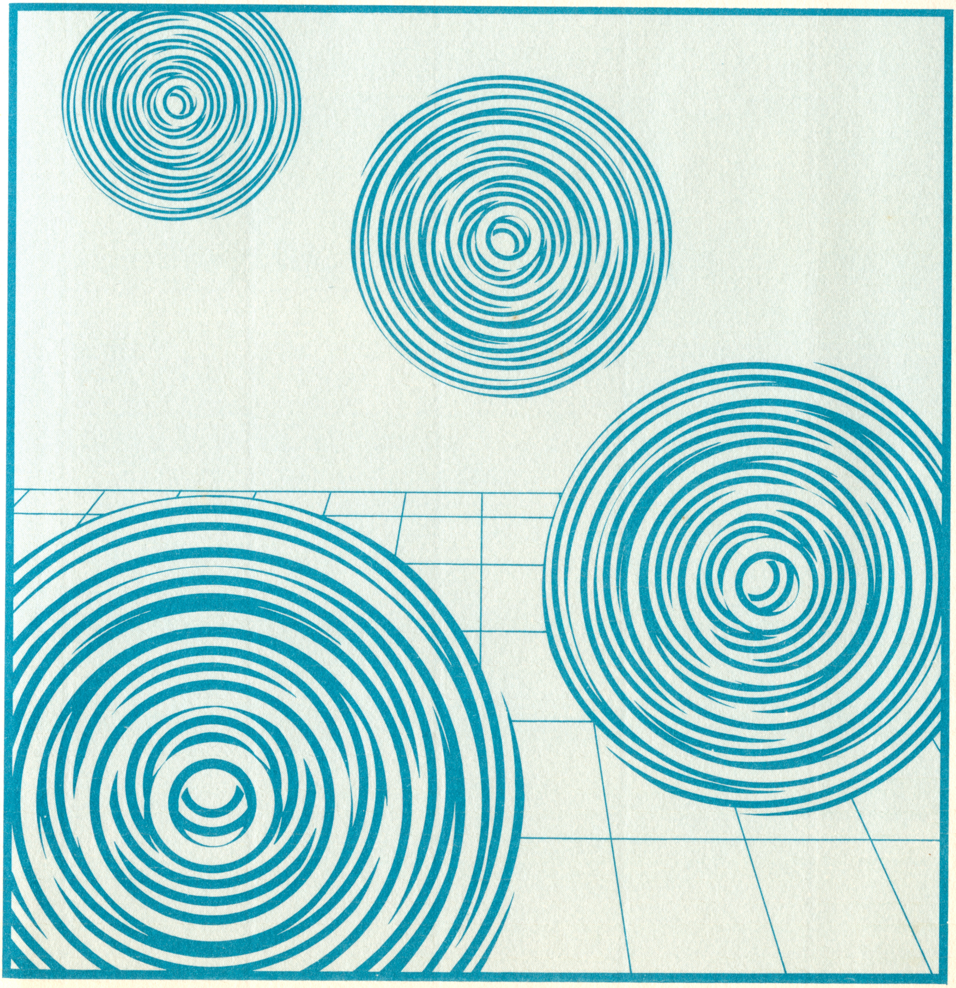
マイコンBASICマガジンの大人気特集
「移植テクニック・マスター大作戦」が、
ついに一冊の本になりました。
おなじみのベーマガオールスター総出演です。
異機種間のBASIC移植テクニックを、
おもしろおかしく、わかりやすく解説!!

PART 1
入門編
BASIC
大研究

各機種
BASIC
比較研究!!

パソコン移植テクニック

監修 丹治佐一

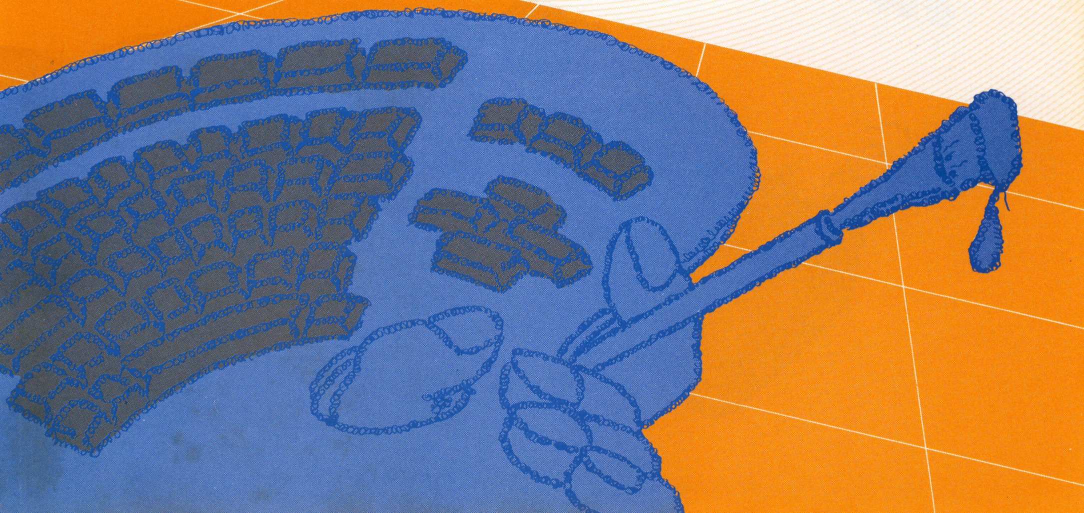


各機種
BASIC
比較研究!!

パソコン移植テクニック

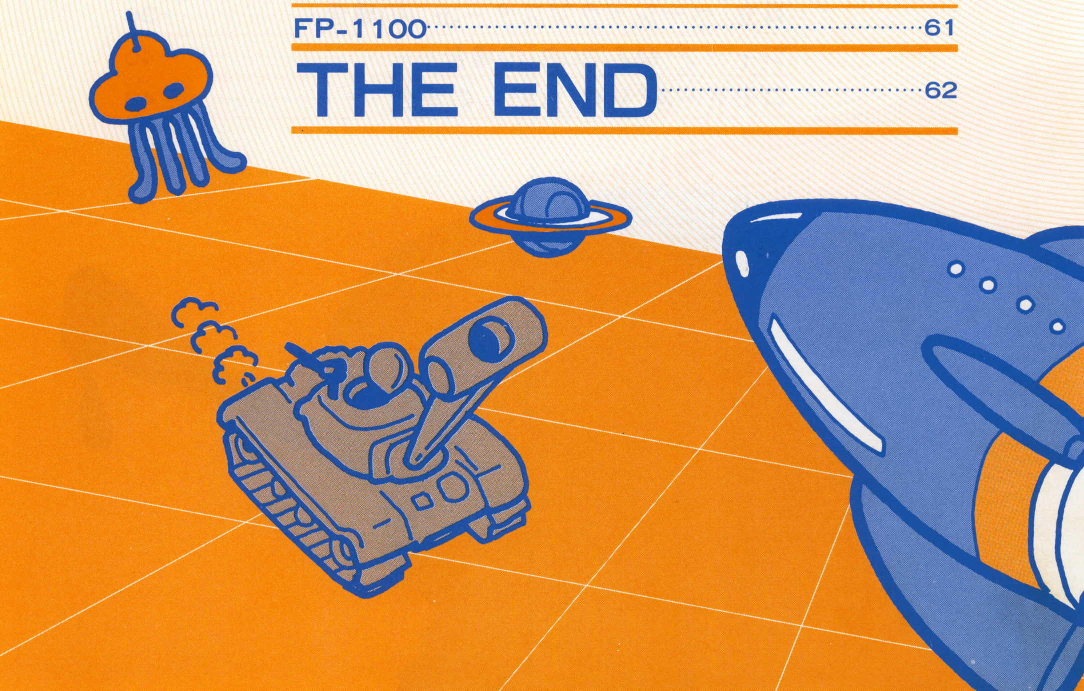
PART 1
入門編
BASIC
大研究

監修 丹治佐一



PART 2

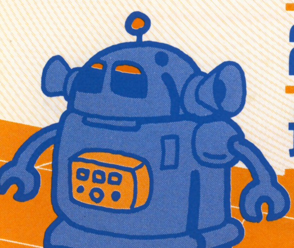
BASIC



PART 3

PART 4





はじめに



丹治佐一

　BASIC言語は簡単な英字による命令でプログラムできるため，パソコンを動かしたことのない人でも，すぐにパソコンを動かせるようになります。
　そんなBASIC言語を塔載したパソコンが数多く出回っています。この移植テクニックでは，各機種のために作られたプログラムを少しでも生かすために，別の機種への移植テクニックを紹介します。
　紹介するにあたって登場する人物がいます。

Dr.D：おなじみデバッグ博士
編：少しはマイコンがわかる編集部員
影：有名な影の声
編集長：中年ながらマイコンをマスターしようとガンバッテいる

という人々です。どうぞよろしく！

丹治佐一



人気機種のBASIC始動と初期設定



方言のあるBASIC言語

編：編集長が食事に行っている間にゲームを入れて楽しむとするか。ポツ，ポツ，ポツ…(キーを押す音)

影：何してるんですか？弁当片手に。

編：某誌におもしろそうなゲームが載ってたんだ。編集長が戻ってくるまでに入れようと思って。

影：あっ！編集長が戻って来た！

編：オッと，某誌をかくさねば！

影：ワッハッハ！冗談，冗談ですよ。そのあわてた顔ったらなかったなー。

編：コラッ！冗談言ってないで手伝ってくれ！

Dr.D：　何騒いでいるのじゃ。

編：パソコンにプログラムを入れているんです。

Dr.D：ホーウ。めずらしく仕事をしていると思ったら

某誌に載ったゲームを入れているのかね。

編：それを言わないでください。

影：博士も手伝ってもらえませんか？

Dr.D：どれ，プログラム・リストを見せてみなさい。

影：（博士に某誌を手渡す）

Dr.D：フムフム，おもしろそうなゲームじゃな。

編：そうでしょ。某誌ながら，なかなかのものを載せてます。

（ガチャ！）ドアが開いた音

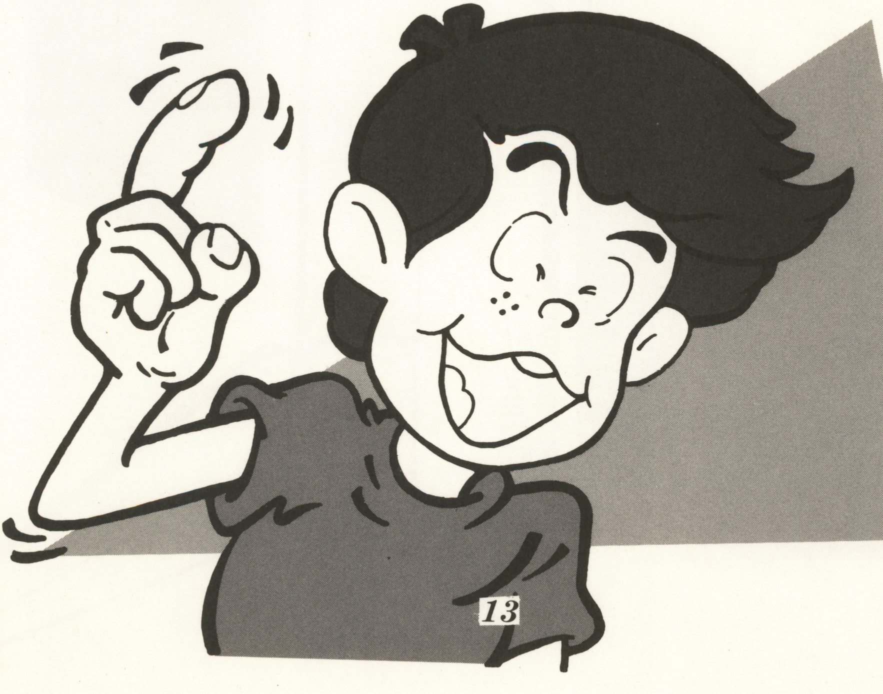
編集長：ああー，うまかった。あの立喰いそばもなかなかの味だったな。明日も行ってみるか。

編，影：あっ！…編，集，長！

編集長：某誌がどうのこうのと聞えたのだが，某誌がどうかしたのか？

編，影：いえ，何でもありません！

Dr.D：ところでだ，君たちが入れているプログラムだが，何の機種のプログラムかをチェックしたのか？



PART 1

編：いえ，ＢＡＳＩＣでプログラムが作られていたので特に気にせず入れているんですが。

影：機種を調べる必要があるんですか？

Dr.D：君たちは，知らんのか？

編集長：博士，みんなに分るように説明していただけませんか？

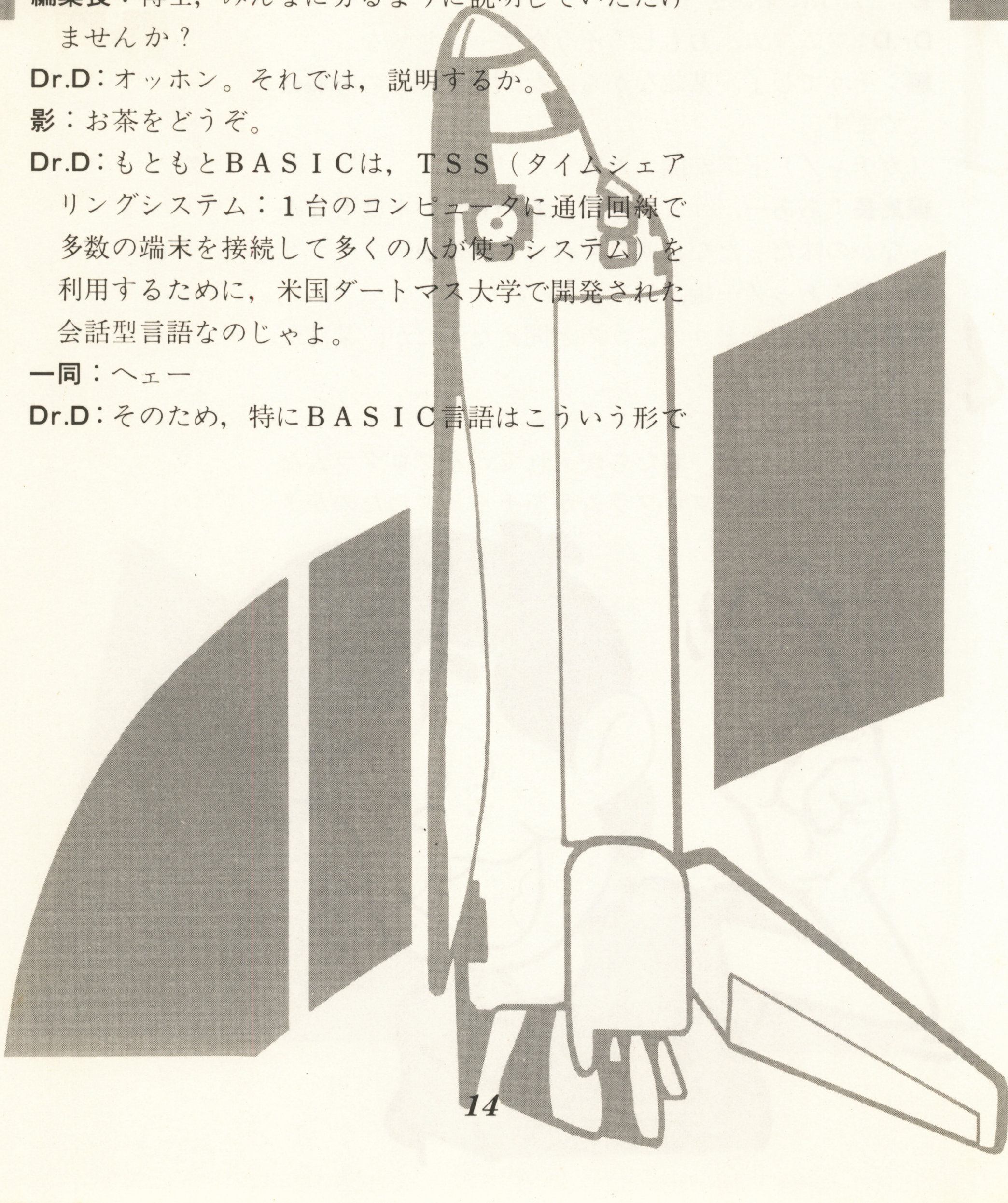
Dr.D：オッホン。それでは，説明するか。

影：お茶をどうぞ。

Dr.D：もともとＢＡＳＩＣは，ＴＳＳ（タイムシェアリングシステム：１台のコンピュータに通信回線で多数の端末を接続して多くの人が使うシステム）を利用するために，米国ダートマス大学で開発された会話型言語なのじゃよ。

一同：ヘェー

Dr.D：そのため，特にＢＡＳＩＣ言語はこういう形で





なければならないという規格もなく，その後，使う人たちが使いやすいように機能をどんどん拡張していったのじゃよ。

一同：ナルホド！

Dr.D：つまり，各社，各機種で，こんな機能は，この命令にしようと勝手に決めてＢＡＳＩＣの機能を拡張したのじゃよ。

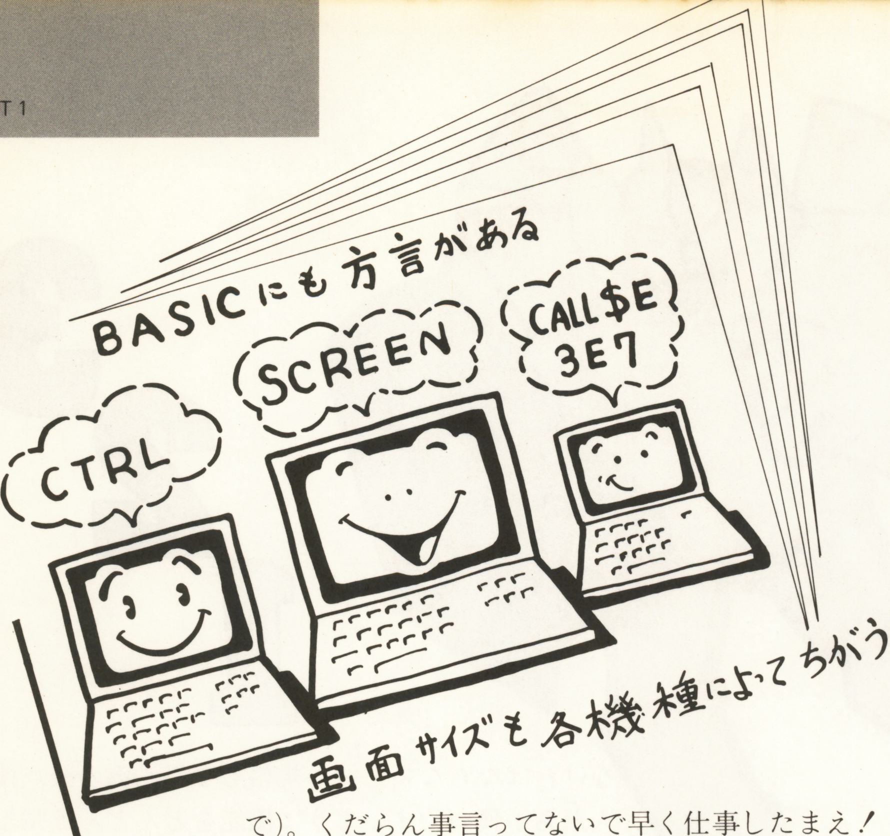
編：ということは，同じＢＡＳＩＣ言語であっても，機能の差や独断と偏見によって違う部分があるということですか？

Dr.D：その通りじゃよ。

編集長：だから，某誌に載っているおもしろいゲームなんか入れようと思っちゃいかんよ！

影：あれ！どうして，某誌におもしろいゲームが載っていることを知っているんですか？

編集長：ゴホッ，ゴホッ（痛い所を突かれたという顔



で）。くだらん事言ってないで早く仕事したまえ！

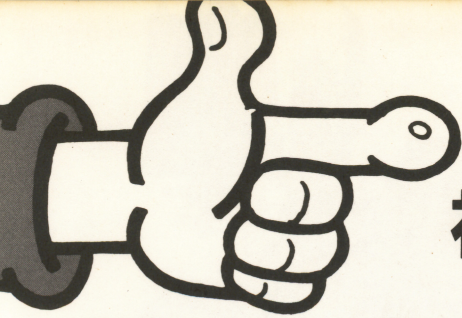
Dr.D：まだ話は終わっとらんぞ！

編：結論は，プログラムを入れようとするときは，その機種のために作られたプログラムでなければ正しく動かない，ということですね。

影：だから，機種を確かめたかを聞いたのですか？

Dr.D：その通りじゃよ。ＢＡＳＩＣ言語には，大なり小なり方言があることを覚えておきたまえ。





初期設定

編：博士！ＢＡＳＩＣ言語の違いなどを知ることができれば，たとえ機種が違ったプログラムでも生かせるのではないでしょうか？

Dr.D：いい所に気が付いたな。

影：あれっ！僕たちが登場しているのは，それを読者のみなさんにお知らせするためではなかったんですか？

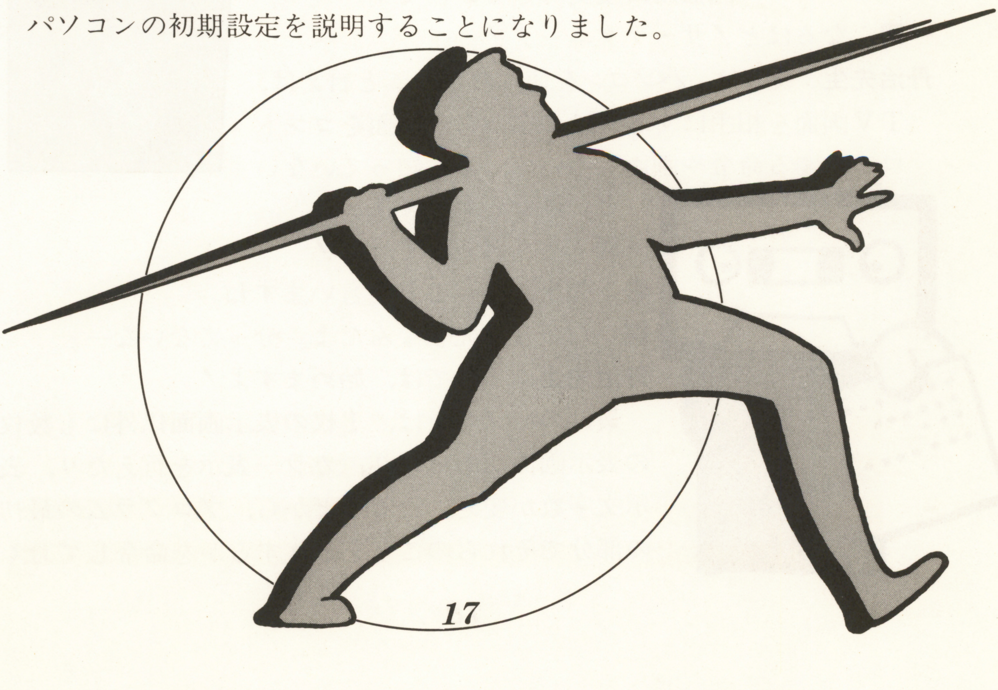
編集長：コラッ！話の腰を折るような事を言うな！

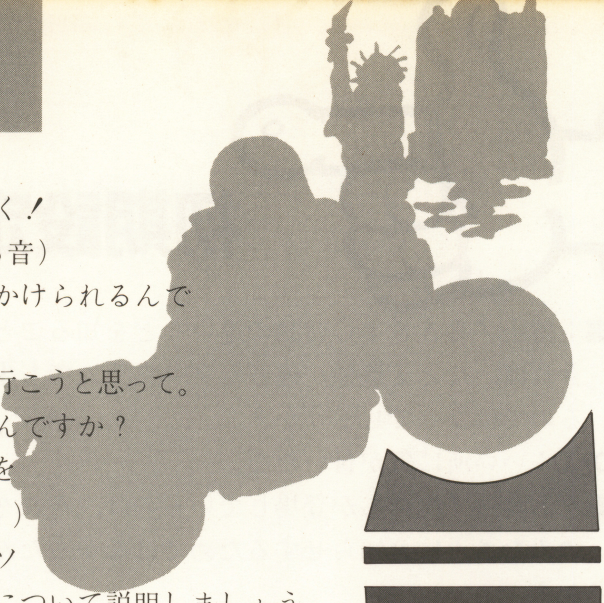
影：すいません！

Dr.D：そうくるだろうと思って，有名な先生を呼んでおいた。もう来る頃じゃ。

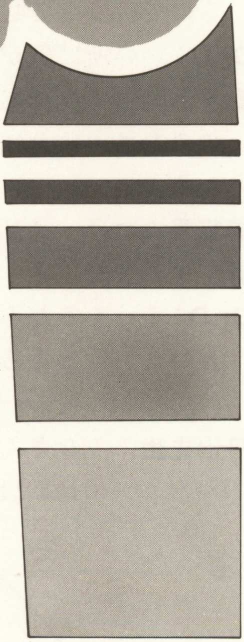
（コン，コン，ガチャ！ドアを開けて入って来る）

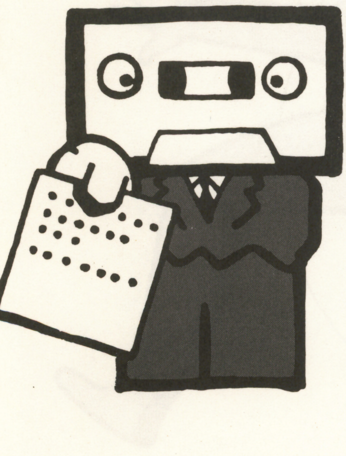
丹治先生：みなさん，こんにちは！博士にたのまれてパソコンの初期設定を説明することになりました。





よろしく！
一同：こちらこそよろしく！
（ガチャ！ドアを開ける音）
影：博士！どちらかへ出かけられるんですか？
Dr.D：立食ソバを食べに行こうと思って。
編：それで先生を呼んだんですか？
Dr.D：まあ，まあ…（ドアをあわてて閉め，出ていく）
丹治先生：それでは，パソコンの画面の初期設定について説明しましょう。
影：どうして，画面の初期設定を知る必要があるんですか？
丹治先生：それは，これを知ることで，どんな画面で動作させているかを知ることができますから，他のパソコン用に作られたプログラムであっても，自分のパソコンの画面用に変更できるわけです。
一同：なるほど！ザ・テクニック！
丹治先生：それに，パソコンを使ったゲームと言えば，ＴＶ画面を相手にするわけですから，画面をコントロールする命令や画面について，よく知っていないといけないわけです。





一同：そう言えばそうだ！
影：先生，いいことを言いますね。
編：だから，先生なんだよ。分ってないなー。
丹治先生：それでは，始めますよ！
最近のパソコンは，１枚の表示画面以外にも数枚の表示画面があったり，カラー表示を行えたり，表示文字数が変えられることから，プログラムの最初の部分でそれらのことをどうするかを命令しておく



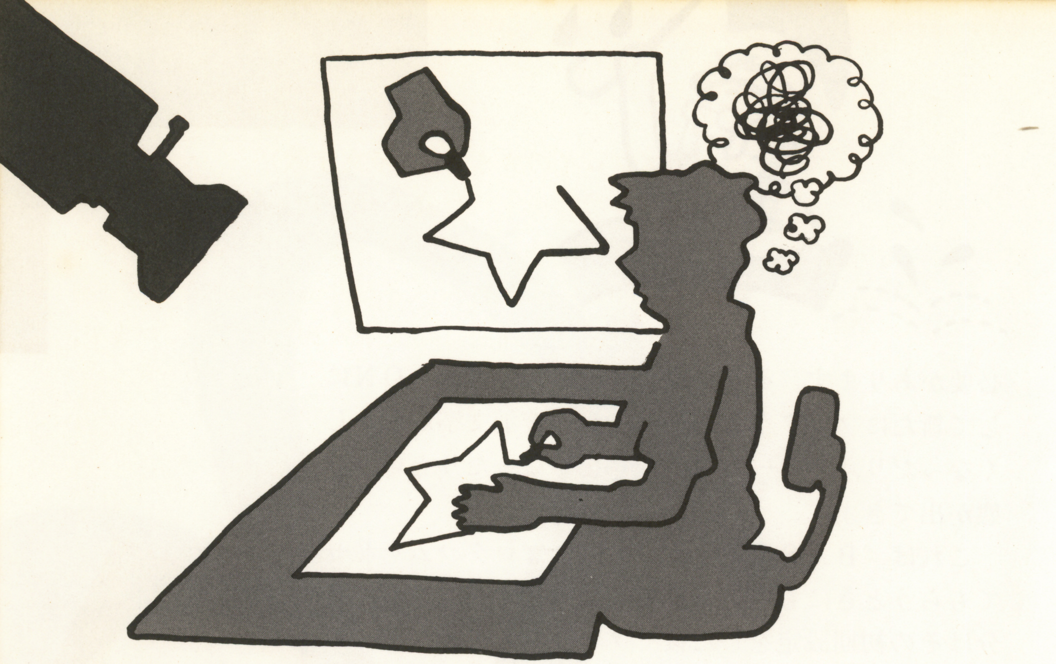
必要があります。そうしておかないと，電源をＯＮにして新たにプログラムを動かしたら表示文字がおかしくなったり，カラー表示ができなくなったりという状態が出てきます。

これは〝ＢＡＳＩＣマガジン〟にプログラムを載せてもらうときにもいえる重要なことです。つまり，自分はその初期設定をした後でプログラムを動かしているため，本に載ったプログラム・リストにはその初期設定の命令が入っていない場合もあるからです。知らずに打込んで実行させるとおかしな画面になってしまうというトラブルが発生してしまいます。



まず読取りたい各機種の画面コントロールなど

それでは，各機種について説明していくことにしましょう。その前に注意があります。たとえ画面をコントロールする命令などを自分のパソコン用の命令にしたとしても，マシン語を使っていたり，画面の内容を読んでいたりするとそう簡単には直すことができません。



たとえば表示文字を横に80文字表示するプログラムは，自分のマシンが40文字しか表示できなければ80文字表示しているプログラム自体を40文字用にすべて変更しなければいけないわけです。その他いろいろありますが注意して変更してください。

各機種に共通していえることは，プログラムの大きさ(メモリの使用量)にも注意することです。メモリが足りなくてはいくら変更できてもお話になりません。

NEC「PC-8001」の場合



★パワーONからBASICのスタート

特に問題はない。パワーONですぐBASICが動きだす。

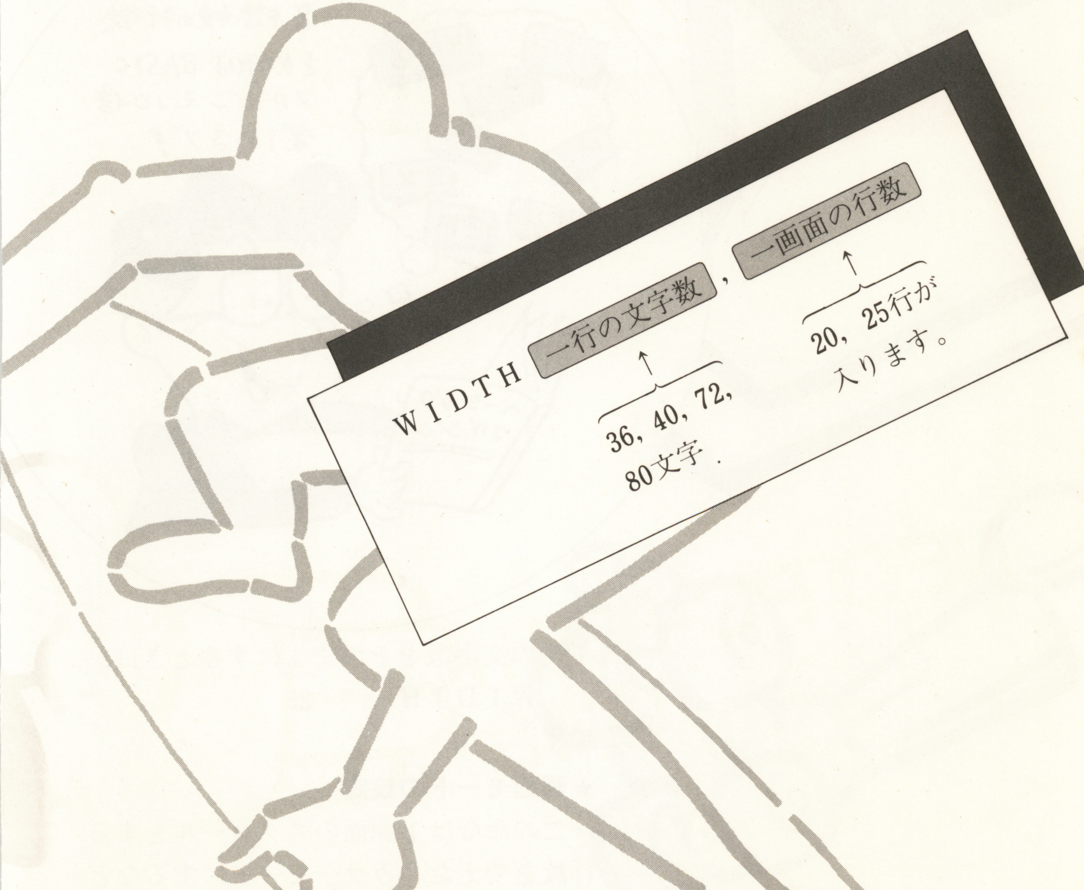
★ディスプレイの大きさ(表示文字数)

表示文字数は横１行に36，40，72，80文字が選べ，縦方向の行数は20，25行が選べます。つまり横36文字×20行～80文字×25行の８とおりあるわけです。

パワーＯＮ時の画面は40文字×20行になっています。

★表示文字数を変える命令

一画面に表示できる文字数を変える命令です。





前と同じ値のところは省略することができます。たとえば一行の文字数だけを80文字にして，行数はそのままでおきたいときは，

WIDTH　80

でOKです。また，行数を25行にして



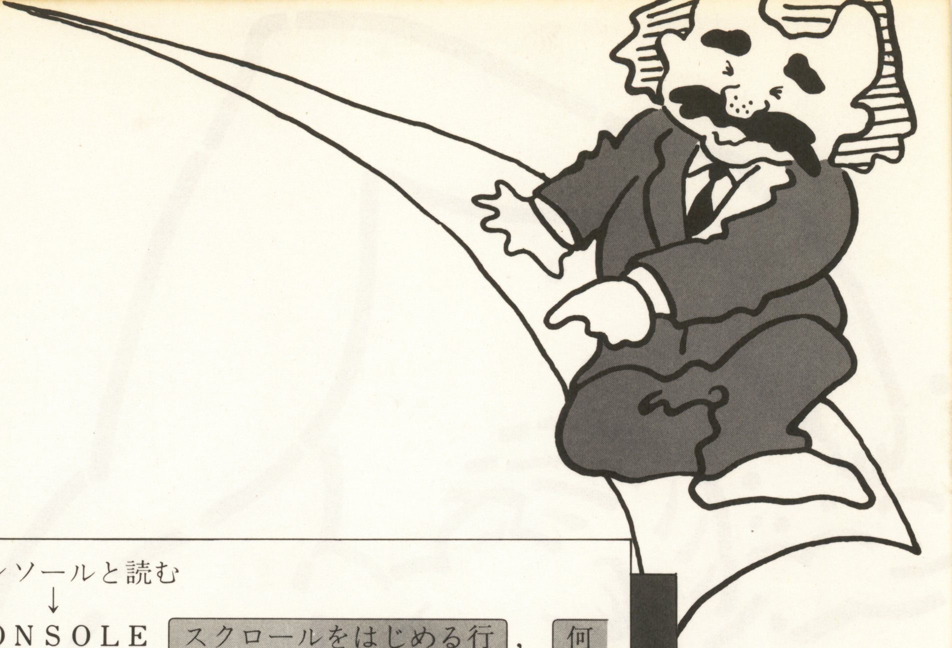
1行の文字数をそのままにするときは，

WIDTH　,　25

です。

★画面モードの設定

この命令は1画面のスクロールをする行数を変えたりカラーモードにするなどの設定を行います。



コンソールと読む
↓
CONSOLE スクロールをはじめる行, 何行分スクロールさせるか?, ファンクション・キーを表示させるか?, カラーモードにする?
↑
行数を入れる



スクロールをはじめる行と何行分スクロールさせるかはパワーONのとき1画面分の大きさになっています。つまり，0行目から25行スクロールさせるようになっています。

ファンクション・キーの表示は1で「表示」0で「表示しない」です。また，カラーモードは1で「カラーモード」，0で「白黒モード」です。

たとえばファンクション・キーを表示させずにカラーモードにするには

CONSOLE ,, 0, 1

となります。WIDTH同様に変え



ない値は省略できるわけです。

★カラー命令

文字の色指定やグラフィック・モードの設定を行います。

カラーと読む
↓
COLOR 色 , ヌルキャラクタ , グラフィック・スイッチ

ヌルキャラクタ：画面を消したときぬられるキャラクタの番号，キャラクタモードのとき 32，グラフィック・モードのとき 0 です。

グラフィック・スイッチ：1 グラフィック　0 キャラクタモード

色は第1－1表に示します。

■第1－1表　PC-8001のカラーコード

CONSOLEコマンドでカラーモードになっている場合，0～7まで入力可能で，0～7に対応している色はつぎのようになる。

0 － 黒	4 － 緑
1 － 青	5 － シアン
2 － 赤	6 － 黄
3 － マゼンタ	7 － 白

白黒モードでは，画面のディスプレイ・モードを変えることができる。

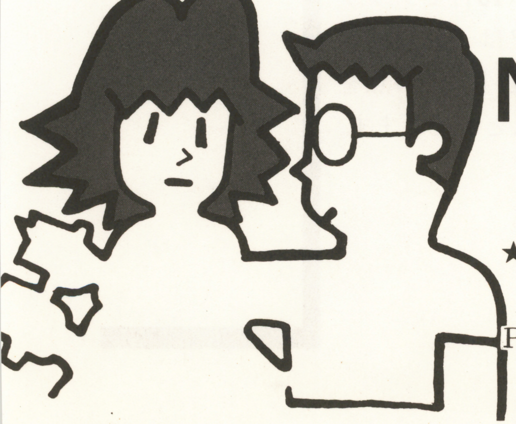
0 － ノーマル	4 － リバース

★画面クリア

画面を消す方法はキーボードから CLS キーを押すかプログラム中からは，

PRINT　CHR$(12)

を実行します。





NEC「PC-6001」の場合

★パワーONからBASICのスタート

パワーONすると〝HOW MANY PAGES？〟と聞いてきます。これは

何ページ分の画面を使いますかということですから，メモリのゆるす限り最大限のページ数を使いたいのであれば RETURN キーのみ押します。

これでBASICはスタートします。

★ディスプレイの大きさ（表示文字数）

各モードによって表示文字数やグラフィックの点の数が変わります。

●モード１〈テキスト（文字）モード〉

文字のみの画面で１行32文字×16行表示できます。パワーＯＮ時の画面です。

●モード２〈セミグラフィック・モード〉

文字と四角い点が表示できます。

文字は，１行32文字×16行でグラフィックは64（横）×64（縦）ドットです。

●モード３〈低分解能グラフィック・モード〉

モード２とくらべて細かなグラフィックが表示できます。文字は１行16文字×16行でグラフィックは 128（横）×192（縦）の点を打つことができます。

●モード４〈高分解能グラフィック・モード〉

モード３よりさらに細かな点を表示できます。文字は32×16行でグラフィックは 256（横）×192（縦）ドットの点を打つことができます。



★表示文字数を変える

表示文字数を変えるということはPC－6001では表示モードを変えることに相当します。

スクリーンと読む

SCREEN モード，書込むページ，表示するページ

※PC－8001同様値を変えない所は省略できます。

モード：1～4までのモード

書込むページ：点を打ったり表示する画面のページ

表示するページ：実際目で見るページ数は本体のままで2ページ（2画面）

※ROM，RAMカートリッジを付けると4ページ使用可能になる。







(例)モード２にして書込むページ，見るページを１ページにします。

```
SCREEN  2, 1, 1
        ↑  ↑  ↑
        │  │  └─見るページ
        │  └─書込む(実行するページ)
        └─モード2(セミグラフィックにする)
```

(注)モード３以上はプログラム実行中でないと画面が変わりません。

(例)モード３の画面を見続ける(ページは１)






```
10 SCREEN 3, 1, 1
20 GOTO 10
```

★画面コントロール

画面のスクロールの指定，ファンクションキーの表示，キーボードを押したときの音のON／OFFを行う命令です。

CONSOLE スクロール開始行, スクロールする行数, ファンクションキーの表示, キー音のON／OFF

ファンクションキーの表示：
1で表示
0で表示しない

キー音のON／OFF：
1で音がする
0で音がしない

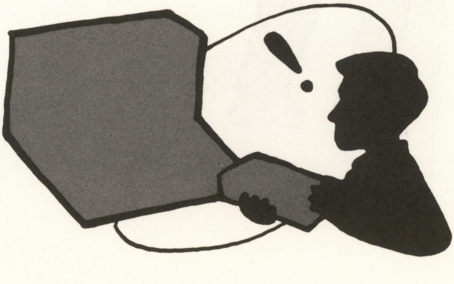
※PC－8001同様，値を変えない所は省略できます。

㊎ファンクション・キーを表示させず，キー音もさせない。

```
CONSOLE ,, 0, 0
```

★色を変える

色は各モードによって指定できる数が変わります。



COLOR　　文字の色，文字の背景の色，　色
　　　　　　　↑　　　　　↑　　　　↑
表では　　　　"F"　　　　"B"　　　"C"
と表わしています。



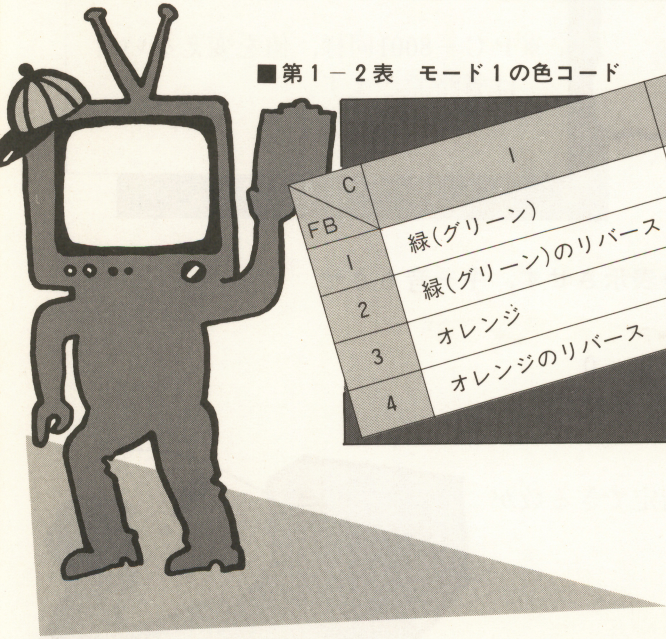
★画面を消す

命令は

CLS　です

あるページを指定して消す場合，SCREEN 命令を使って，書込むページをそのページにします。

各モードの色コードは**第1－2表～第1－5表**に示します。



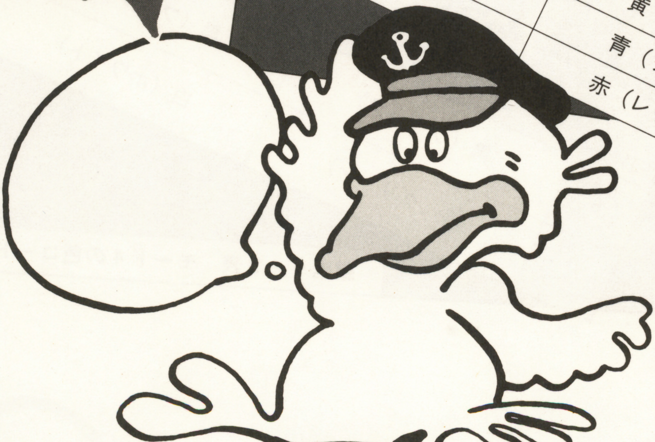
■第1－2表　モード1の色コード

FB ＼ C	1	2
1	緑(グリーン)	オレンジ
2	緑(グリーン)のリバース	オレンジのリバース
3	オレンジ	緑(グリーン)
4	オレンジのリバース	緑(グリーン)のリバース



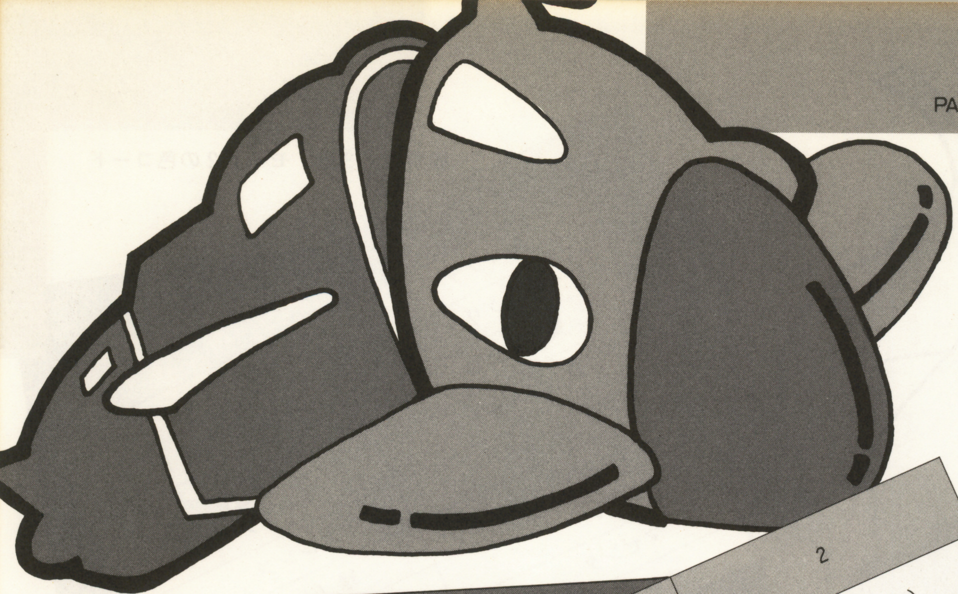
■第1－3表　モード2の色コード

FB＼C	1	2
0	黒（ブラック）	黒（ブラック）
1	緑（グリーン）	白（ホワイト）
2	黄（イエロー）	シアン
3	青（ブルー）	マゼンタ
4	赤（レッド）	オレンジ
5	白（ホワイト）	緑（グリーン）
6	シアン	黄（イエロー）
7	マゼンタ	青（ブルー）
8	オレンジ	赤（レッド）



■第1－4表　モード3の色コード

FB＼C	1	2
1	緑（グリーン）	白（ホワイト）
2	黄（イエロー）	シアン
3	青（ブルー）	マゼンタ
4	赤（レッド）	オレンジ



FB \ C	1	2
0	黒（ブラック）	黒（ブラック）
1	緑（グリーン）	白（ホワイト）

■第1－5表　モード4の色コード



富士通「FM-8」の場合

★パワーONからＢＡＳＩＣのスタートまで

本機もパワーＯＮと同時にＢＡＳＩＣがスタートします。

★ディスプレイの大きさ（表示文字数）

１画面，横１行40文字または80文字で縦20行または25行を選べます。パワーＯＮ時は40文字×25行になっています。40×20行～80×25行の４とおりあります。

★表示文字を変える

命令はＰＣ－8001と同じＷＩＤＴＨです。

ＷＩＤＴＨ　［１行の文字数］，［縦の行数］

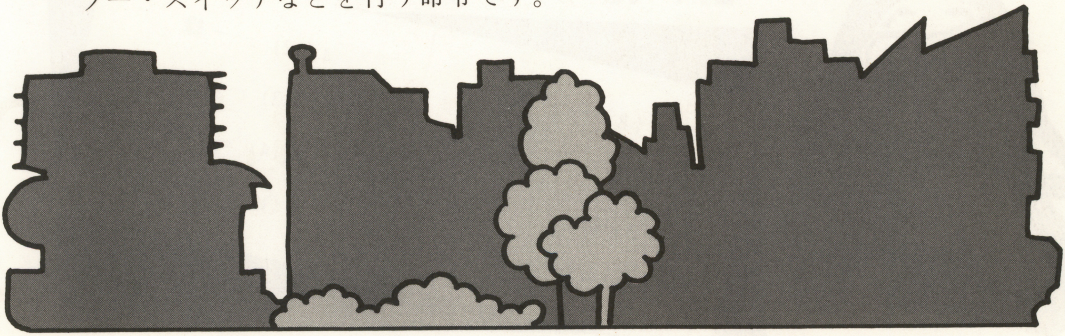
１行の文字数：40，80文字
縦の行数：20，25行

㊎80文字×25行の表示画面にする

ＷＩＤＴＨ　80，25

★画面コントロール

スクロールの行やファンクション・キーの表示，カラー・スイッチなどを行う命令です。



CONSOLE スクロール開始行, スクロールする行数, ファンクション・キー表示スイッチ, カラー・スイッチ

ファンクション・キー表示スイッチ：
- 1 表示
- 0 表示しない

カラー・スイッチ：
- 1 カラーモード
- 0 カラーOFF

※パワーON時は0〜20行までがスクロールするようになっている。

★色を指定

色指定はCOLOR文で行います。

COLOR 文字の色, 背景の色

パワーON時には COLOR 7, 0 つまり文字が白で背景が黒です。

色コード表は**第1－6表**に示します。



■第1－6表　カラーコード表

0 (8) —— 黒	4 (12) —— 緑
1 (9) —— 青	5 (13) —— 水色
2 (10) —— 赤	6 (14) —— 黄
3 (11) —— 紫	7 (15) —— 白

★画面を消す。

命令はCLSです

色を指定するCOLOR文で背景の色を変える時にもCLSを実行します。

㊎文字の色を白にし，背景を青にする

```
10  COLOR  7, 1
20  CLS
```

松下「JR-100」の場合



★パワーONからBASICのスタートまで

パワーONですぐBASICがスタートします。

★ディスプレイの大きさ(表示文字数)

画面の大きさは横32文字×24行です。

★画面コントロール

機能	画面の表示モード，オーバーフロー処理モードを選択します。
書式	OPTION {CMODE0｜CMODE1}｜{OVF0｜OVF1}
要点	CMODE0：反転文字モードを選択。

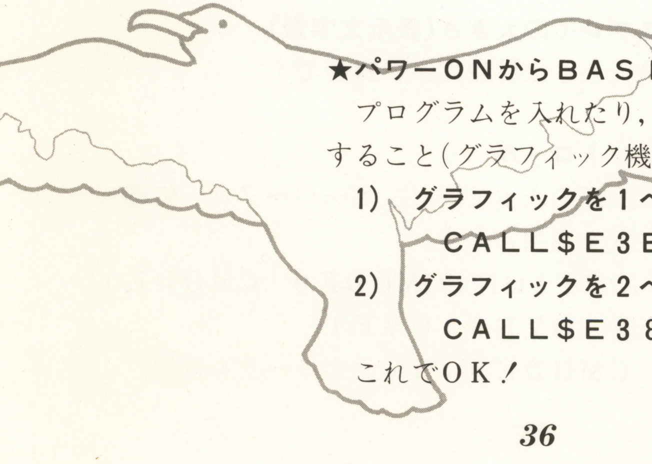
キャラクタコード表のすべての文字とそれの反転文字が表示できます。

ＣＭＯＤＥ１：ユーザ定義文字モードを選択します。キャラクタコード表とユーザ定義文字コード表のすべての文字が表示できます。

ＯＶＦ０：オーバーフローが発生したら直ちにプログラムの実行を中断するオーバーフロー検出モードを選択します。

ＯＶＦ１：オーバーフローが発生しても引き続きプログラムを実行するオーバーフロー非検出モードを選択します。

日立「ベーシックマスターJr」の場合



★パワーＯＮからＢＡＳＩＣのスタートまで

プログラムを入れたり，作る前に必ず次の文を実行すること(グラフィック機能を使うときのみ)。

1) **グラフィックを１ページ(１画面)だけ使う場合**

   ＣＡＬＬ＄Ｅ３Ｅ７

2) **グラフィックを２ページ(２画面)とも使う場合**

   ＣＡＬＬ＄Ｅ３８３

これでＯＫ！

★ディスプレイの大きさ(表示文字数)

1画面の横1行32文字×縦24行で高分解能グラフィックは横256×縦192ドット表示できます。

★文字数を変える

文字数は固定です。



★表示モードの切替え

テキスト・モード，グラフィック・モード（ページ1，2）の3モードがあります。

1） テキスト・モード(文字モードにする)

POKE$EFE0，0

2） グラフィック1ページ選択

POKE$EFE0，$C0

3） グラフィック2ページ選択

POKE$EFE0，$CC

★色指定

カラーにするにはカラー・アダプタが必要です。

POKE＄E892，1 POKE＄E890，㋑	⬅プログラムの最初に行う（カラーRAMをオープンさせる）。

㊟カラー画面をスクロールさせる場合POKE＄E892，0を実行した後行う。

POKE＄E892, 1でカラー使用指定の後，POKE＄E890, ⊗で色を指定します。これは他機種のCOLOR文と同じでおまじないだと思っておけばよいでしょう。

POKE文で⊗の部分に書込む色コードは，つぎのように指定します。

POKE＄E890,　＄○○……**16進数2桁**

文字背景色（上位桁）　文字色の色コード（下位桁）

文字色と文字背景色は別に指定できます。

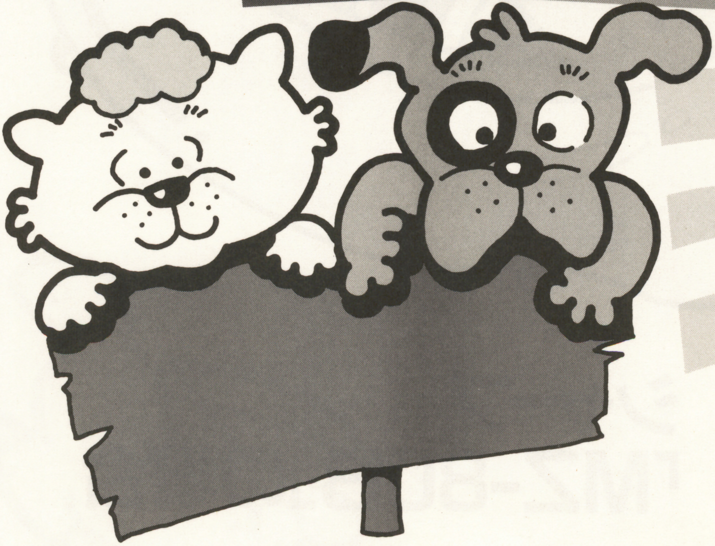
色コードは，**第1－7表**のようになっています。



■第1－7表　ベーシックマスターJr.の色コード

16進数		色	16進数		色
0	8	黒	4	C	緑
1	9	青	5	D	シアン
2	A	赤	6	E	黄
3	B	マゼンタ	7	F	白

8～Fを指定するとブリンク(点滅)します。

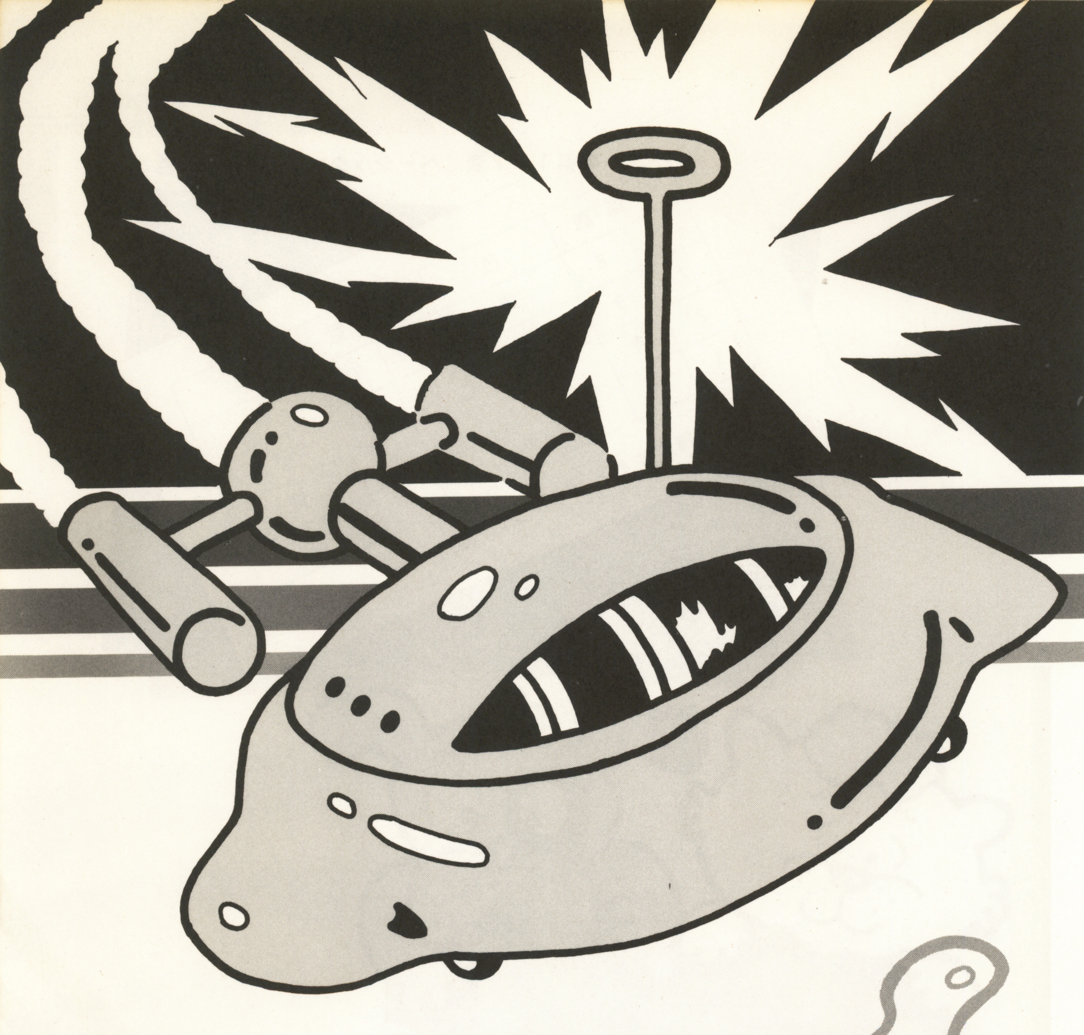


★画面を消す

1)　TEXTモードは，CLEARまたはCLRです。
　　　↑文字

2)　グラフィック1ページ目は，CALL＄E38D

3)　グラフィック2ページ目は，CALL＄E39C



シャープ「MZ-80B」の場合

★パワーONからBASICスタートまで

ＢＡＳＩＣインタプリタを本体のＲＡＭに入れる必要がありますので，パワーＯＮのつぎに巻取られている〝ＢＡＳＩＣインタプリタ・テープ〟「ＳＢ－5520」

などをカセット・デッキの中に入れ，フタを閉めればＯＫ。ロード後，自動的にＢＡＳＩＣがスタートします。

★ディスプレイの大きさ（表示文字数）

１画面，横40文字または80文字，縦25行の２とおりの表示文字数があります。

★画面コントロール

スクロール行，表示文字数，白黒反転の設定を行う命令です。

ＣＯＮＳＯＬＥ　**スクロール開始行，スクロールする行数，表示文字数，白黒反転**

表示文字数	白黒反転
１行	ノーマル
40文字にする	（通常の画面）
Ｃ40	Ｎ
80文字にする	リバース
Ｃ80	（白黒反転）
	Ｒ



★グラフィック・コントロール

グラフィック画面を設定する命令です。

グラフと読む

　　↓

GRAPH　**書込むページ，表示するページ，グラフィック・エリアのクリア，グラフィック・エリアをぬりつぶす**

○書込むページ──→I **ページ数**

　　　　　　　↑1～2

○表示するページ→O **ページ数**

　　　　　　　↑1～2

{ 0のとき表示しない
  1 2(21)のとき両ページ表示

○グラフィック・エリアのクリア──→C

○グラフィック・エリアをぬりつぶす→F

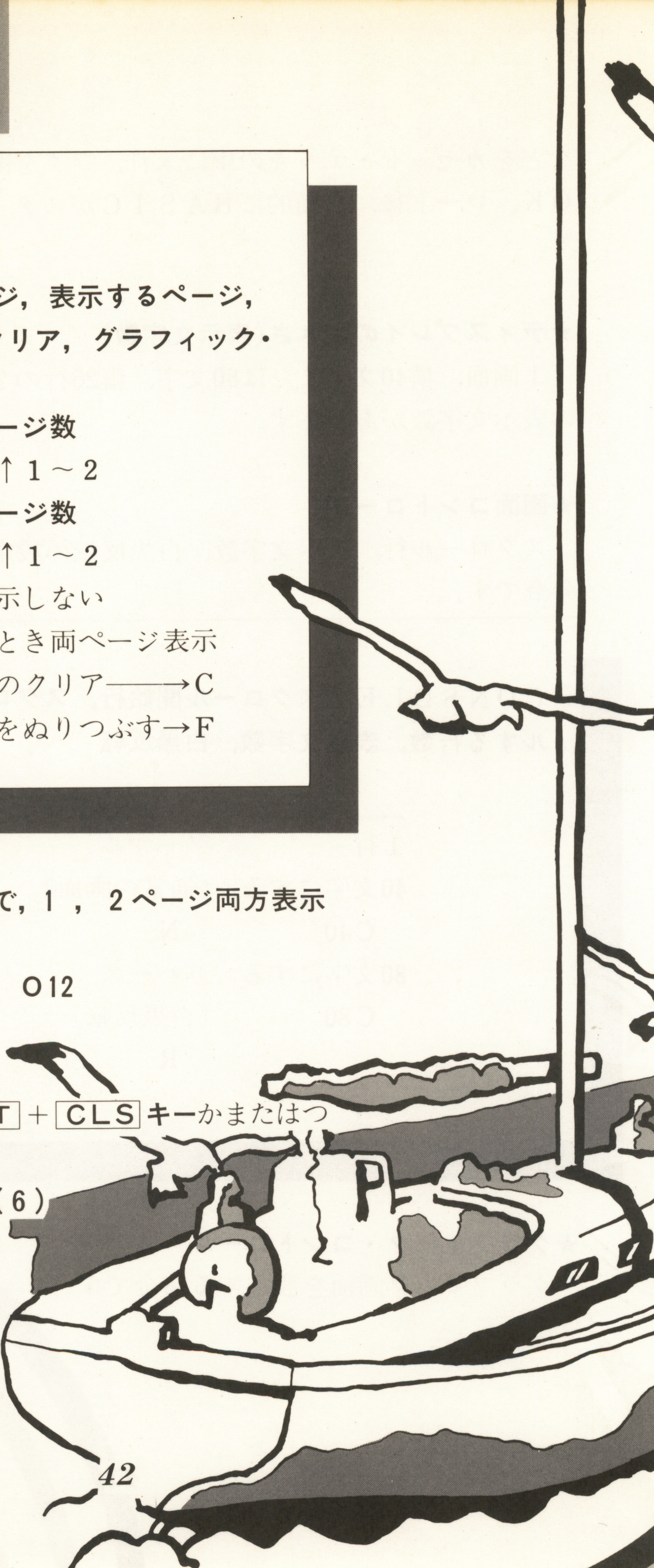
(例)　2ページ目に書込んで，1，2ページ両方表示させる。

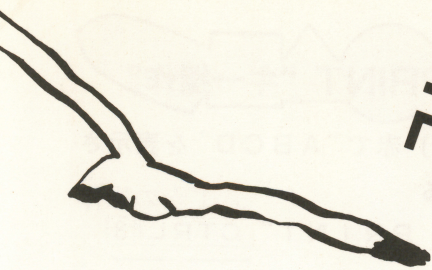
```
GRAPH  I2, O12
```

★画面を消す

画面を消すには SHIFT ＋ CLS **キー**かまたはつぎの命令を実行します。

```
PRINT  CHR$(6)
```





コモドール「VIC-1001」の場合



★パワーONからBASICのスタートまで

電源ONですぐBASICがスタートします。画面上に使用できるメモリの量も表示されます。

★ディスプレイの大きさ(表示文字数)

文字数は横22文字×23行です。スーパーエクスパンダVIC－1211を使用すると176×160ドットの高分解能グラフィックも表示できます。

★文字数を変える

文字数は固定です。



★スクリーン・コントロール

コントロールは，画面クリア，カーソル移動，カラー指定など，キー操作によって行い，プログラム中に行うときはプリント文中でこれらの操作を行うと反転文字として画面に表示されます(動作はしない)。

プリント文中に入れたそれらの操作の実行はそのプログラムを実行したときに行われます。

PRINT “キー操作”

（例）赤で“ＡＢＣＤ”を表示させる

　　　　　　　　　↓赤の操作
PRINT“[CTRL][3]
ABCD”　[¥]の反転マークとして表示される

（ガチャ！ドアが開く音）

Dr.D：もう終わったかな？

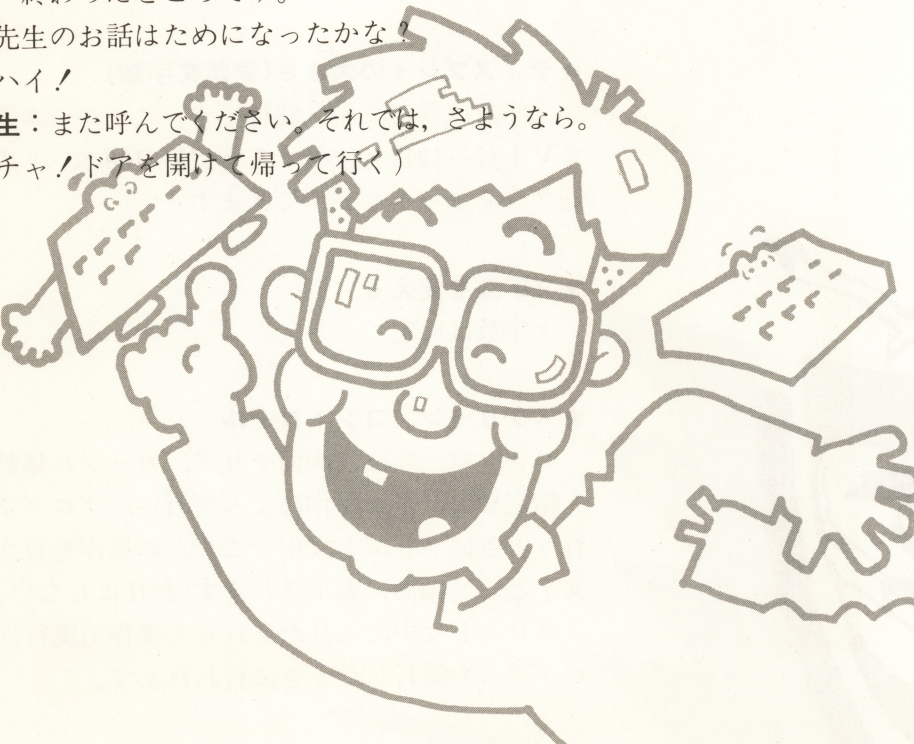
編：今，終わったところです。

Dr.D：先生のお話はためになったかな？

一同：ハイ！

丹治先生：また呼んでください。それでは，さようなら。

（ガチャ！ドアを開けて帰って行く）



■第1－8表　VIC-1001の色コードなどの命令表

〔　〕内の文字は，コモドールキーとSHIFTキーをおしたときにでる文字

コマンド	キー操作	画面にでてくる文字
ファンクション・キー1	“f1”	[illegible]　〔[illegible]〕
ファンクション・キー2	“SHIFT＋f1”	[illegible]　〔[illegible]〕
ファンクション・キー3	“f3”	[illegible]　〔[illegible]〕
ファンクション・キー4	“SHIFT＋f3”	[illegible]　〔[illegible]〕
ファンクション・キー5	“f5”	[illegible]　〔[illegible]〕
ファンクション・キー6	“SHIFT＋f5”	[illegible]　〔[illegible]〕
ファンクション・キー7	“f7”	[illegible]　〔[illegible]〕
ファンクション・キー8	“SHIFT＋f7”	[illegible]　〔[illegible]〕
(1) 画面クリア	“SHIFT＋CLR HOME”	[illegible]　〔[illegible]〕
(2) ホーム・ポジション	“CLR HOME”	[illegible]
(3) カーソル上	“SHIFT＋CRSR(↑↓)”	[illegible]　〔[illegible]〕
(4) カーソル左	“SHIFT＋CRSR(←→)”	[illegible]
(5) カーソル下	“CRSR(↑↓)”	[illegible]
(6) カーソル右	“CRSR(←→)”	[illegible]
(7) 一字追加	“SHIFT＋INST DEL”	[illegible]　〔[illegible]〕
(8) 反転文字	“CTRL＋R”	[illegible]
(9) 反転文字解除	“CTRL＋0”	[illegible]　〔[illegible]〕
(10) ブラック	“CTRL＋1”	[illegible]　〔[illegible]〕
(11) ホワイト	“CTRL＋2”	[illegible]
(12) レッド	“CTRL＋3”	[illegible]
(13) シアン	“CTRL＋4”	[illegible]　〔[illegible]〕
(14) マジェンタ	“CTRL＋5”	[illegible]　〔[illegible]〕
(15) グリーン	“CTRL＋6”	[illegible]
(16) ブルー	“CTRL＋7”	[illegible]
(17) イエロー	“CTRL＋8”	[illegible]

（注）“の印はダブルクォーテーション。＋の記号は前のキーといっしょに押す。

BASIC命令の省略形の巻



Dr.D：この章ではみんなにプログラムを入力してもらおうと思う。そこで君たちの使命だが，いかに短かい時間でそれらを入力することができるかどうかにある。例によって，君もしくは君の仲間が，エラーにとりつかれ，あるいは発狂しても当局は一切関知しないからそのつもりで。なお，このテープは自動的に消滅する。成功を祈る。

編：アッ！ア～ァッ！せっかく読者が送ってくれたテ

ープを――。どうしてくれるんです!!

Dr.D：心配するでない。ここに，そのプログラム・リストが，あるではないか。

影：なるほどね。

編：なるほどねじゃないですよ。こんなに長いプログラムを，いったい誰が入力するんですか。

Dr.D：もちろん，君たちさ。

影：エーッ！そんなアホな！

編：まさか～？どこかにマスター・テープをかくしてあるんでしょう？意地悪なんだから，Dr.Dは。

編集長：いや，さっき確認したのだが，間違いなく本物のマスター・テープだったよ！

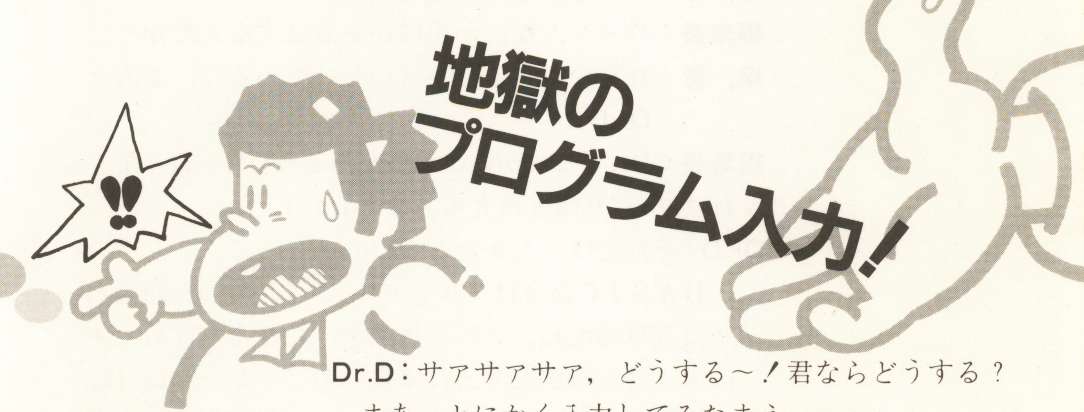
編，影：ア然！

Dr.D：君たちもしつこいね。さっさとやらないと，読者にしかられるぞ。

編集長：こうなったらしかたない。いっしょうけんめい入力するか。

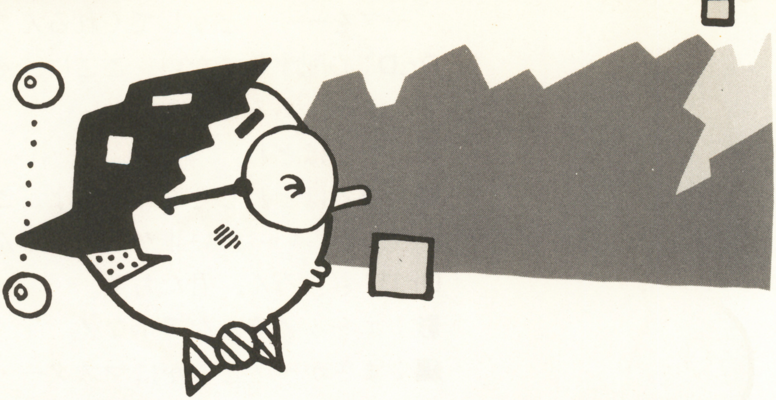
編，影：ポカーン！

地獄のプログラム入力！



Dr.D：サァサァサァ，どうする～！君ならどうする？まあ，とにかく入力してみたまえ。

編：ブツブツ！ヘン！わかりましたよ。入力すればいいんでしょ。



影：(編の顔色をうかがって)じ，じゃあ，私はラ製編集部に帰りますから。

編：ジロッ！（と憎しみのまなざしで）

影：…………わかった，手伝うよ。手伝えばいいんだろう。

編集長：ところで，さきほどDr. が，できるだけ短い時間で入力すると言っていたのは，どういうことなのだろうかな。

編，影：そんなこと知りませんよ～。

編集長：ウ～ン，なにか手段があるはずなんだが―。

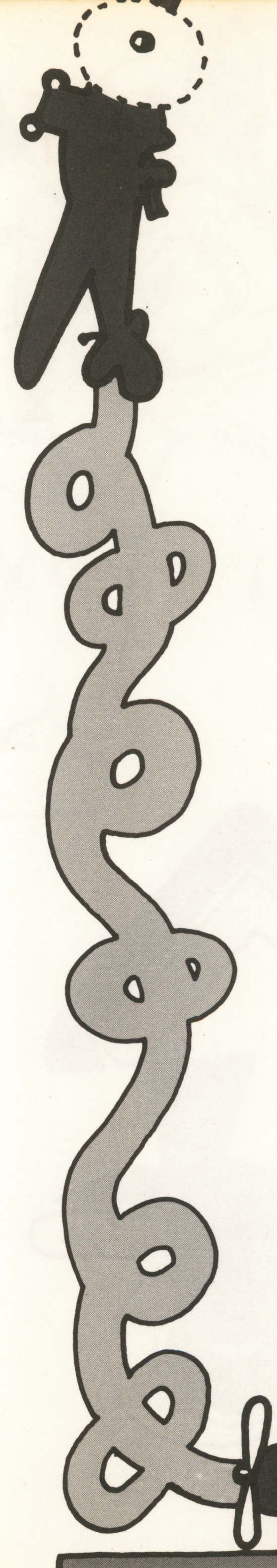
編，影：知らないもんね～（いじけているな，2人とも。：Dr.D)。

編集長：Dr.！なにかいい方法があるのなら，教えてくれませんか(ほとんど苦しまぎれに)。

Dr.D：それでは，しかたがない。

　BASIC命令はアルファベットをいくつか組合わせた英単語だね。この英単語を，できるだけ打たずにすませる方法を考えれば良いのじゃよ。それには，それぞれのBASICに準備された，省略形を知っていると便利だ。題して，「各機種省略形を知ろう」だ。

編集部一同：賛成～!!



省略形を覚えよう!!



Dr.D：省略形の用意された機種も多いが，省略形が見あたらない機種もある。それらの機種も含めて，考えてみることにしようか。

編：省略形がわかると，プログラムを入力する手間も省けるのでしょうから，早く，早く教えて下さいよ!!

Dr.D：そうせかすなよ。省略形の話をはじめる前に，省略形のある機種，ない機種を，まず調べてみることにしようか。

影：そんなこと言わないで，早く!!

編集長：影君はしばらく，紙袋でもかぶって，だまっていなさい。

影：ムッ！（としながら，袋をかぶってしまった）

Dr.D：よ～し，袋だたきにしろ!!

編集部一同：ドタ！ポカッ！ゴツン！イテッ！

影：♧，♧，………。

Dr.D：やっとうるさいやつが１人消えたな。あと１人，ジロッ！（と編のほうをにらむ）





編：まさか，そんな…。ネッ！ネッ！はじめましょ。たしか，機種別に見た省略形の有無でしたね。

Dr.D：そうだ，君もなかなかよくわかってきたのう。まず，省略形のない機種はと—。



ZX-81
JR-100
JR-200

Dr.D：この3機種には省略形がない。なぜだか君は知っているかな，編君!!

編：エッ？わかりませんね。まさか会社の方針ではないですよね。

編集長：ワンキー・コマンドがあるから……でしょう？

Dr.D：オッ！なかなか目のつけどころがいいな。その



■第1－9表　省略形①

JR-100/JR-200/ZX-81

JR-100, JR-200は，CTRLキー併用によるワンキーコマンド入力が可能なので省略形は必要ないわけ。
ZX-81は，すべてワンキーコマンド命令を入力するため省略形はない！

とおりじゃ。省略形というのは，プログラム入力時に，できるだけキー操作する回数を減らそうという思想のもとに作られたものだからな。

編：すると，ＺＸ－81のシンクレアＢＡＳＩＣは，全身これ省略形のカタマリといったところですか？

Dr.D：やったね！編君！汚名バンカイのひと言だな。

編：いゃ～，それほどのことではないですよ。

Dr.D：JR-100やJR-200も同様だ。



編：これらの機種は，どうですか？

Dr.D：シャープ系のＢＡＳＩＣとＰＣ－6001は，ＰＲＩＮＴ命令を“？”で代用できるが，それ以外の省略形はないんだ。また，このＰＲＩ

■第１－10表　省略形②

MZ-80K/C/1200, MZ-80B/2000, PC-6001

	予約語	省略形
CRTへのDATA表示	**PRINT**	**?**

ＮＴ命令を“？”で代用する方法は，他のおもだった機種も同様だね。

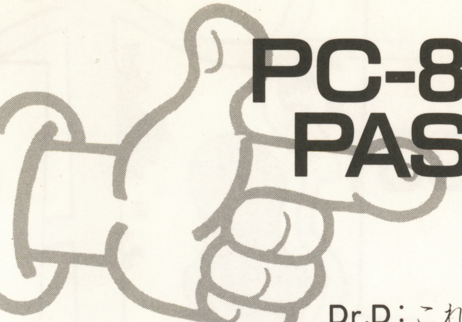
編集長：しかし，なぜ省略形が少ないんですかね？

Dr.D：ワシもよくわからんが，おそらくＢＡＳＩＣインタプリタを，小さくするためではないかな。

編：どういうことですか？

Dr.D：ＢＡＳＩＣインタプリタが省略形をうけつけるには，省略形を解釈できるような構造にしなければならないわけだ。そのための手間もあるし，変換のためにＢＡＳＩＣインタプリタが，大きくなることも考えられるからな。

　つまり，インタプリタを省略形のために大きくせず，同じ大きさで処理内容を充実させようということだと思う。



PC-8801/PC-8001 PASOPIA/HC-20

Dr.D：これらは，ＰＲＩＮＴ文の“？”の他に，ＲＥＭ文の“，”が使えるようになっておる。

編：省略形が少ない理由は，さきほどと同じ理由からでしょう。

Dr.D：だろうな。

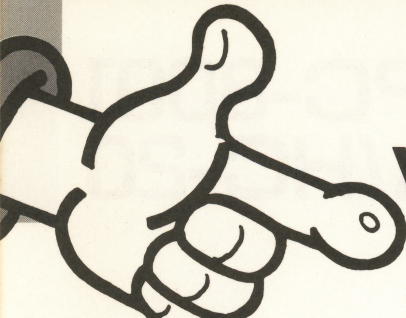
■第１－11表　省略形③

PC-8801/PC-8001, PASOPIA（T-BASIC）, HC-20

	予約語	省略形
CRTへのDATA表示	PRINT	?
コメント	REM	,

編集長：少し弱気になっていますね。

Dr.D：そういうわけではないが，このような内部構造にかかわる内容については，プログラムを開発した人間に直接聞いてみないと，正確なコメントはできないからね。



VIC-1001



編：ＶＩＣに省略形があったのですか？

Dr.D：あるんだとも。それもほぼすべての命令に対してね。

編集長：ホント！どうなっているのですか，いったい。

Dr.D：使い方は，簡単!!ＢＡＳＩＣのそれぞれの命令を，それと識別できる所まで打ちこみ，最後の文字をシフトキーを押しながら打つわけだ。

編：なんかよくわからないですね。

Dr.D：例えば，ＰＲＩＮＴ命令の場合は，ＰとＲで識別できるね。

■第1－12表　省略形④

VIC-1001

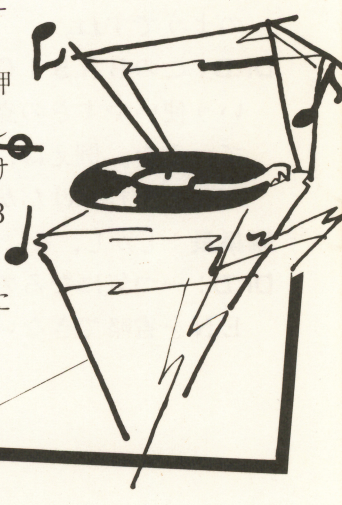
予約語	キー操作	省略形
AND	A Ⓝ	A▨(Aホ)
NOT	N Ⓞ	N□(Nマ)
CLOSE	CL Ⓞ	CL□(CLマ)
CLR	C Ⓛ	C□(Cリ)
CMD	C Ⓜ	C▨(Cへ)
CONT	C Ⓞ	C□(Cマ)
DATA	D Ⓐ	D♠(Dチ)
DEF	D Ⓔ	D□(Dナ)
DIM	D Ⓘ	D□(Dノ)
END	E Ⓝ	E▨(Eホ)
FOR	F Ⓞ	F□(Fマ)
GET	G Ⓔ	G□(Gナ)
GOSUB	GO Ⓢ	GO♥(GOモ)
GOTO	G Ⓞ	G□(Gマ)
INPUT#	I Ⓝ	I▨(Iホ)
LET	L Ⓔ	L□(Lナ)
LIST	L Ⓘ	L□(Lノ)
LOAD	L Ⓞ	L□(Lマ)
NEXT	N Ⓔ	N□(Nナ)
OPEN	O Ⓟ	O□(Oミ)
POKE	P Ⓞ	P□(Pマ)
PRINT	?	?
PRINT#	P Ⓡ	P□(Pメ)
READ	R Ⓔ	R□(Rナ)
RESTORE	RE Ⓢ	RE♥(REモ)
RETURN	RE Ⓣ	RE□(REヤ)
RUN	R Ⓤ	R□(Rユ)

予約語	キー操作	省略形
SAVE	S Ⓐ	S♠(Sチ)
STEP	ST Ⓔ	ST□(STナ)
STOP	S Ⓣ	S□(Sヤ)
SYS	S Ⓨ	S□(Sル)
THEN	T Ⓗ	T□(Tネ)
VERIFY	V Ⓔ	V□(Vナ)
WAIT	W Ⓐ	W♠(Wチ)
ABS	A Ⓑ	A□(Aツ)
ASC	A Ⓢ	A♥(Aモ)
ATN	A Ⓣ	A□(Aヤ)
CHR$	C Ⓗ	C□(Cネ)
EXP	E Ⓧ	E♣(Eリ)
FRE	F Ⓡ	F□(Fメ)
LEFT$	LE Ⓕ	LE□(LEニ)
MID$	M Ⓘ	M□(Mノ)
PEEK	P Ⓔ	P□(Pナ)
RIGHT$	R Ⓘ	R□(Rノ)
RND	R Ⓝ	R▨(Rホ)
SGN	S Ⓖ	S□(Sヌ)
SIN	S Ⓘ	S□(Sノ)
SPC(	S Ⓟ	S□(Sミ)
SQR	S Ⓠ	S●(Sム)
STR$	ST Ⓡ	ST□(STメ)
TAB(	T Ⓐ	T♠(Tチ)
USR	U Ⓢ	U♥(Uモ)
VAL	V Ⓐ	V♠(Vチ)

キー操作は，○のついている文字をシフトモードで押す。つまり，RUN［RETURN］と打つかわりに［R］〔［SHIFT］＋Ⓤ〕［RETURN］と打てばよいわけじゃ！　また，省略形の所のカッコの中は，カタカナモードの時の表示だ。

編集長：ええ，Pの次にRがくるのは，PRINTという語だけですから。

Dr.D：［P］の次に［SHIFT］を押しながら，［R］キーを押す。これで，PRINT命令の省略形のでき上りだ。しかし，GOTOとGOSUBのように，頭の2文字だけで識別できないものもあるね。これは，どちらかを3文字の省略形にする。GOTOとGOSUBなら，［G］［SHIFT］＋［O］，［G］［O］［SHIFT］＋［S］のようにね。



編：へ～っ，便利ですね。



MB-6880L2 6881 6885

Dr.D：この機種も，省略形の得意なマイコンだね。通常よく使う命令については，ほとんど省略できる。

ＩＦ文のＴＨＥＮなどは，次に命令がくる場合は省略してもよい。

編：そうですね。よく使う命令の中で省略できないのは，ＰＯＫＥ命令と，ＦＯＲ－ＮＥＸＴ関係ぐらいのようですね。

Dr.D：このＢＡＳＩＣは，初期のものとは言え，こういう使いやすさの点を考えると，なかなか良くできているな。例えば，他機で[R][U][N]と打つのが，[R]だけですむわけだから。しかし…。

編集長：しかし，なにかまずいことでもあるのですか？

Dr.D：一つ気になるのは，次のようなＩＦ文は，ＴＨＥＮを省略できないんだ。



〈だめな例〉

その1.

IF　A＝B　THEN　PRINTA　IFA
＝B？A

これを省略したければ，次のようにする。

IF　A＝B　PR　A

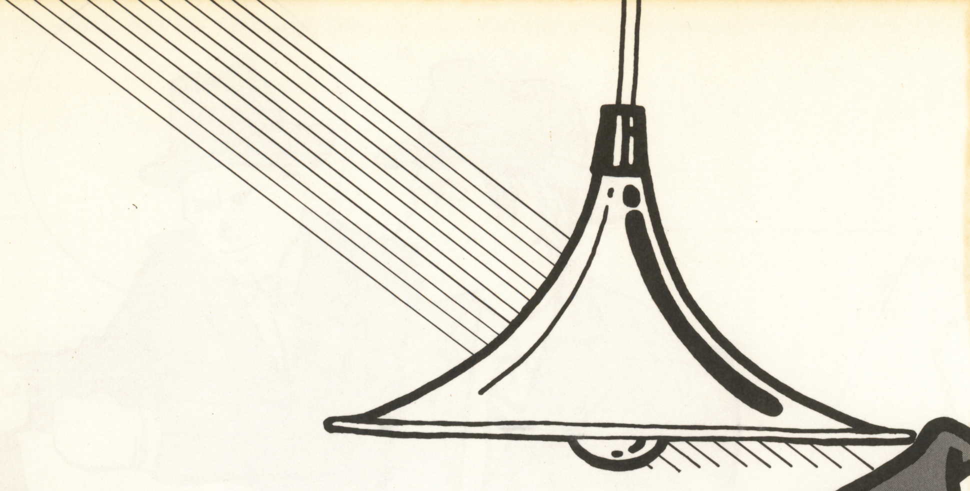
これは，CURSOR＝の省略形についても言えることだ。

その2.

IF　A＝$3FF　THEN　FOR　I＝0
TO10：NEXT　I

➡IF　A＝$3FF　FOR　I＝0　TO10
：NEXT　I

これは，FORのFが，16進数のFとしてとられてしまうためだ。このように，16進のA－Fを頭文字とした単語のときは注意。以上，気をつけてくれたまえ。



■第1－13表　省略形⑤

MB-6880L2/6881/6885

キーワード	省略形	概略内容および使用例等
RUN	R	プログラムの実行
RUN n	Rn	初期状態に設定した後，行番号nから実行
LIST	L	最初から最後までのプログラムリストを一気に出力するBREAKで出力停止
LIST m, n	Lm, n	行番号mからnまでのプログラムリストを出力する
LIST#	L#	プログラムリストをプリンタに出力する
CONTINUE	CONT またはC	STOPおよびBREAKで中断されたプログラムの次の文から続けて実行する
SIZE	S	プログラムの大きさ，変数領域，メモリの残りバイト数の表示
MONITOR	MON	モニターにジャンプする
PRINT	PRまたは？	画面へこのキーワードに続く内容を表示する
PRINT#	PR# または？#	プリンタへこのキーワードに続く内容を出力する
INPUT	IN	キー入力の要求(RUN後キー入力を要求してくるので数値，文字を入れる)
INPUT "ストリング"	IN "ストリング"	文字を画面に表示しキー入力を要求（同上）
LET	LET自体 全てを省略可	変数に数値または文字を入れる
GOTO	GO	ジャンプ
GOSUB	GOS	サブルーチンへジャンプする
RETURN	RET	サブルーチンの最後に入れ，サブルーチンを呼んだ文の次の文へ戻る
ON…GOTO	ON…GO	ONの条件によりとび先の異なるジャンプを行う
OO…GOSUB	ON…GOS	ONの条件によりとび先の異なるサブルーチンへジャンプ
CLEAR	CLR	表示している画面をクリアし，カーソルを画面左上に移す
RANDOMIZE	RNDM	実行のたびごとに乱数の発生を不規則にする
MUSIC	MU	音楽機能の指定
CURSOR	CUR または！	カーソルの制御（例 CUR＝10, 5　カーソルを画面10, 5の位置へ移動する A＝CUR　変数Aにカーソルの座標を入れる）
CURSOR$	CUR$ または！$	カーソルの位置にあらわされている文字を表わす

FM-8
MB-6890
MB-6891



Dr.D：２機種（３機種かな？）とも，マイクロソフト開発の6809用ＢＡＳＩＣが使用されているため，命令体系も，省略形の使いかたも類似している。ＭＥＲＧＥ命令の省略形が，ベーシックマスターには付いていないが，他はすべて同じだ。英文字を省略した形に，“．”をつけて作るわけだ。

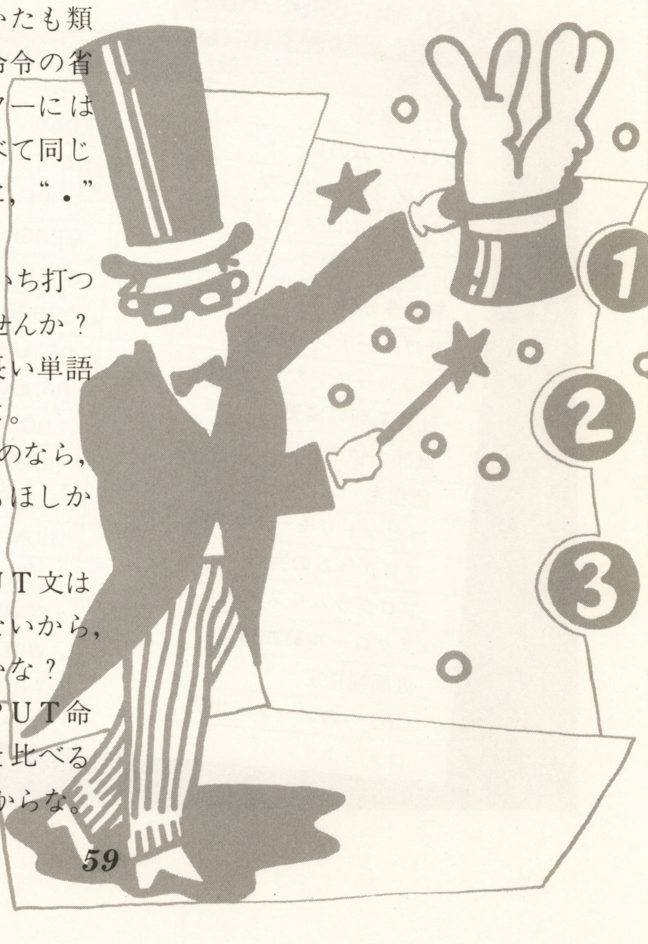
編：だけど，ピリオドをいちいち打つのは，めんどうだと思いませんか？

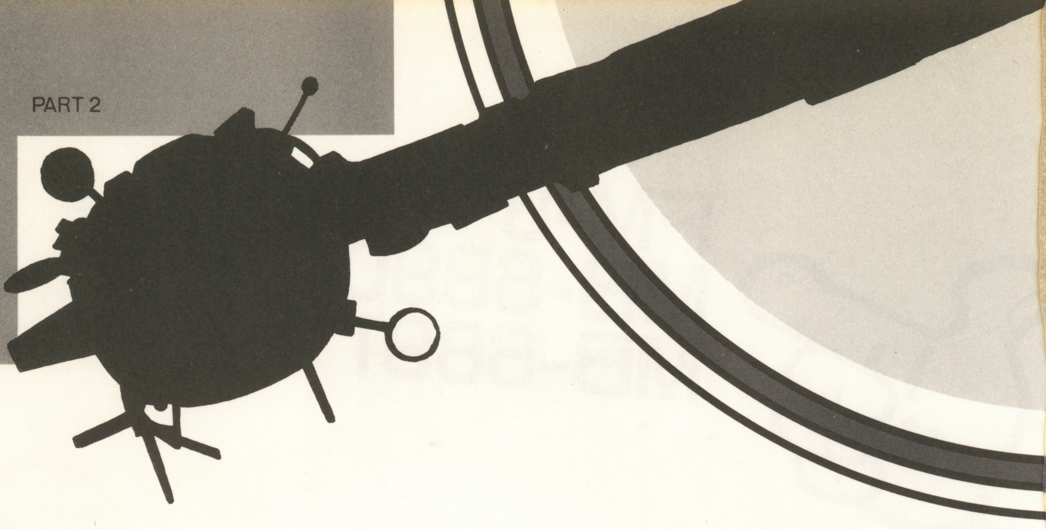
Dr.D：そうだな。しかし，長い単語を打ちこむよりは楽じゃよ。

編：どうせ省略形を用意するのなら，ＩＮＰＵＴ命令の省略形もほしかったのではないですか？

編集長：いや編君，ＩＮＰＵＴ文は意外に使用する割合が少ないから，つけてないんじゃないのかな？

Dr.D：そのとおりだね。ＩＮＰＵＴ命令は，ＰＲＩＮＴ命令などと比べると，使用する回数が少ないからな。

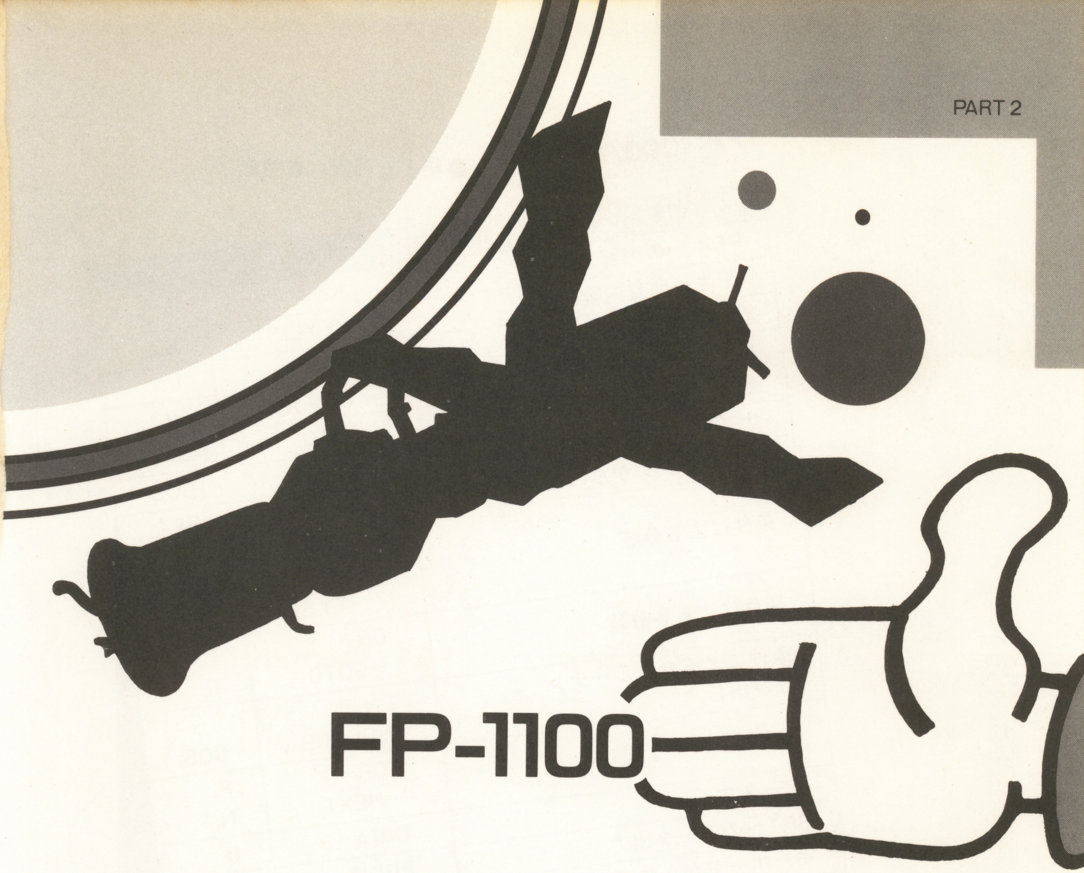




■第1－14表　省略形⑥

FM-8, MB-6890/6891

機　　能	予約語	省略形
実行の継続	CONT	C.
プログラムのロード	LOAD	LO.
プログラムのセーブ	SAVE	SA.
プログラムのスキップ	SKIPF	SK.
プログラムのマージ	MERGE	ME.
無条件分岐	GOTO	GO.
サブルーチン分岐	GOSUB	GOS.
サブルーチンからの復帰	RETURN	RET.
乱数系列の変更	RANDOMIZE	RNDM.
画面の位置指定	LOCATE	LOC.
色指定	COLOR	COL.
カセットリモートのON，OFF	MOTOR	M.
プログラムの実行	RUN	R.
プログラムリスト表示	LIST	L.
スクロール範囲指定	CONSOLE	CONS.
画面幅指定	WIDTH	W.
CRTへのデーター出力	PRINT	?
コメント	REM	'



FP-1100

Dr.D：ＦＰ－1100も，前の２機種と同様な省略形の作りかたをするのだな。ただ，ＬＯＣＡＴＥ命令など使用の度合いが高いものの省略形がなかったり，ＦＯＲ～ＮＥＸＴの省略形があったり，多少作り方に疑問は残るが。

編：そうですね。３文字のＦＯＲをＦ．と２文字にするよりは，ＬＯＣＡＴＥをＬＯＣ．と４文字にしてくれた方が助かりますね。

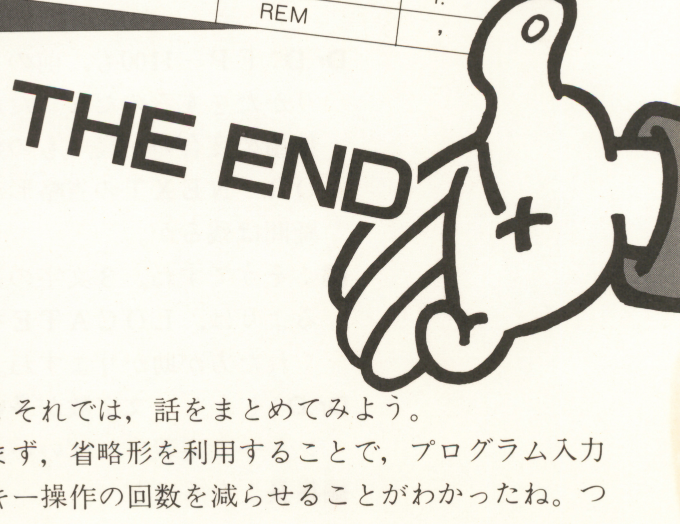
Dr.D：しかし，文字数が多いだけなら，ファンクションキーで定義すればいいんだからな。

編集長：なるほど…。

FP-1000/1100

■第1－15表　省略形⑦

機　　能	予約語	省略形
PFキーの内容定義	KEY	K.
PFキー定義内容の表示	KEYLIST	K.L.
CRTへプログラムを表示	LIST	L.
CRTへ使用変数名,配列名を表示	VARLIST	VL.
無条件分岐	GOTO	G.
サブルーチン分岐	GOSUB	GOS.
サブルーチンからの復帰	RETURN	RET.
条件による分岐	IF ～THEN ～ELSE	— T. E.
条件による多分岐	ON ～GOTO	— G.
条件によるサブルーチン呼び出し	ON ～GOSUB	— GOS.
指定回数による繰り返し	FOR ～NEXT	F. N.
データ文	DATA	D.
CRTへのデータ出力	PRINT	P.
キーボードからのデータ入力	INPUT	I.
コメント	REM	'



Dr.D：それでは，話をまとめてみよう。
まず，省略形を利用することで，プログラム入力のキー操作の回数を減らせることがわかったね。つ

まり，慣れてくれば，かなりプログラムを早く入力できるということだね。また，かくされた省略形を，君たちは知っているかな？

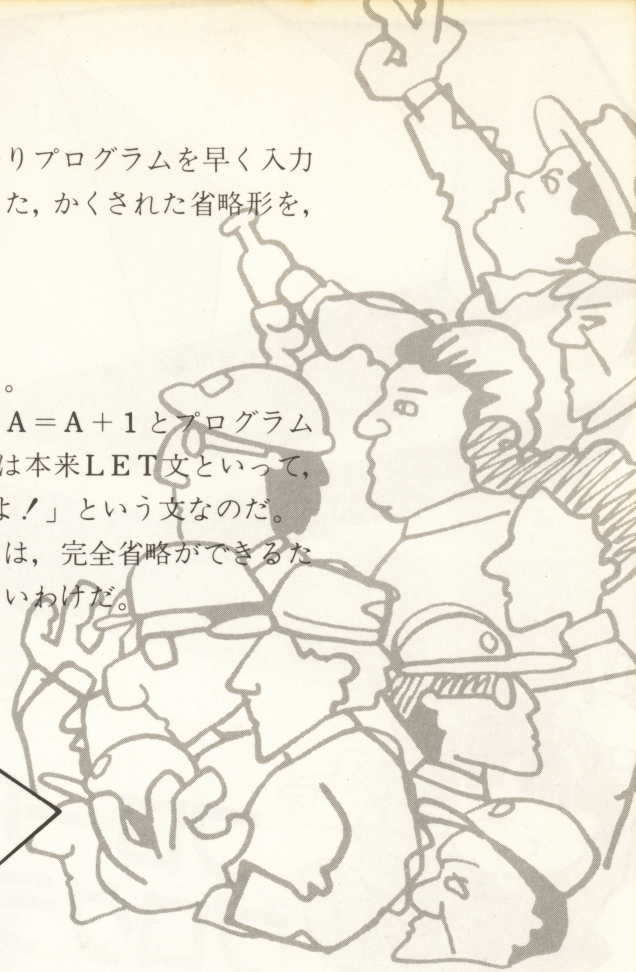
編：編集長，わかりますか？

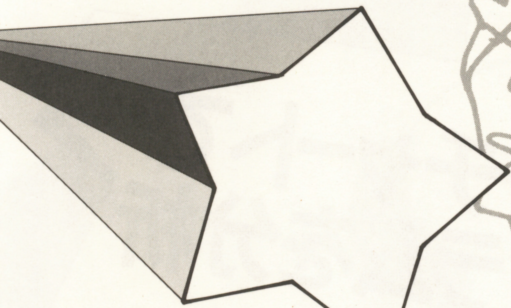
編集長：わからないね〜。

影：わたしも全然わかんない。

Dr.D：数式じゃよ！例えば，A＝A＋1とプログラム中に書いてあっても，これは本来LET文といって，「〜という計算をしなさいよ！」という文なのだ。

　ただ，LET文のLETは，完全省略ができるため，普通はこれらを書かないわけだ。





編集部一同：フ〜ン！

Dr.D：それでは，これで「BASIC命令の省略形」の授業を終わりにしよう。

編：待って！待ってください！さっきのプログラムはどうなったのですか。

Dr.D：これだけ教えたんだから，ガンバッテ入力したまえ。

編：………！

影：あ〜あ，……………………………

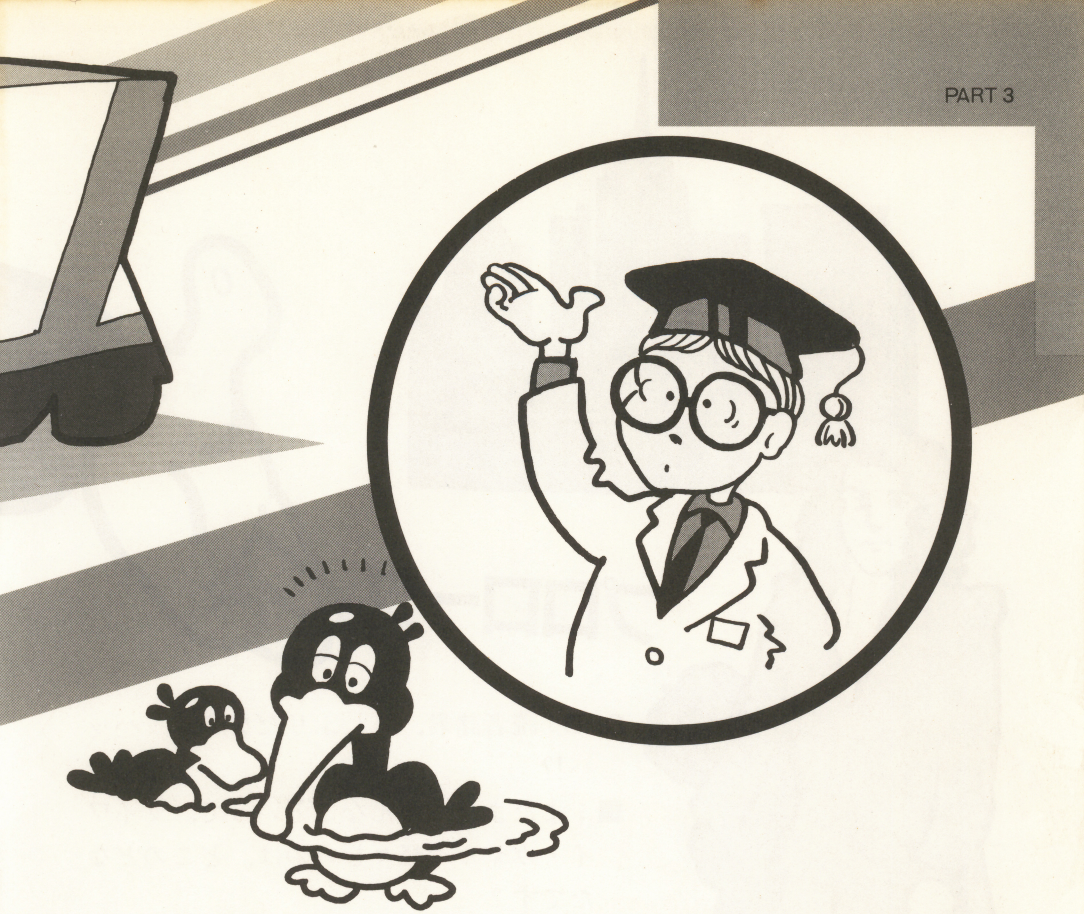
(後日談)

結局編君は，2日間プログラムの入力にあけくれたそうだ(**影**)。



フローチャートでプログラムを分析

さて，ここではフローチャートについてです。フローチャートって知ってますか？　そうです。プログラム作りの基本になる設計図―とでもいいますか，くわしくは本文でふれることにして――えっ？フローチャ



ートが，なぜ移植テクニックに必要なのかって？

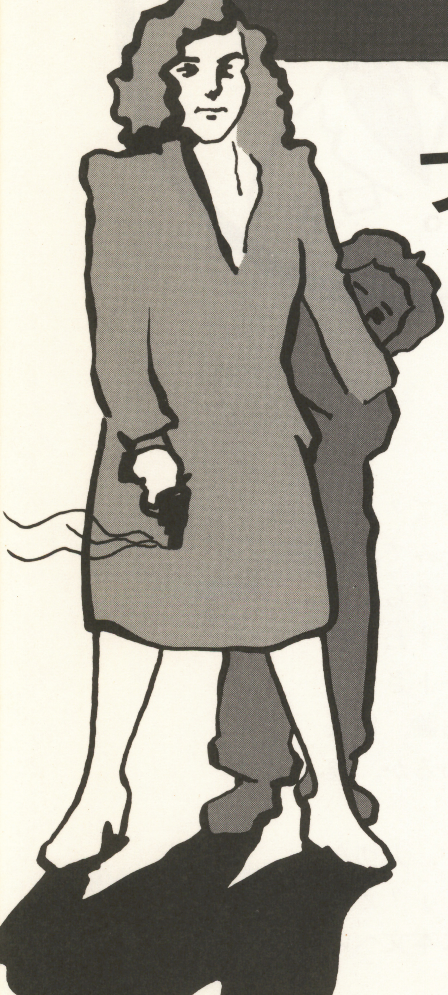
みなさんは，他の人が作ったプログラムをみて，まず何を感じますか？はじめての人ならば，おそらく何が書いてあるかさっぱりわからないと感じますね（当然のことです）。ところが，このフローチャートさえマスターしておけば，自分のプログラムはもち論，他人のプログラムだってひと目で何が書かれているか，理解できちゃうんです。

ちょっとめんどうくさいかもしれませんが，大切なものだってことは，これで十分わかったでしょ？えっ？わかった。それでは，わかっているうちに本文へいってみますか。それ——(**編**)





プロローグ



Dr.D：読者諸君，久しぶりだな。ワッハッハ!?

編：またまた，そんなことといって，ワイワイ・ガヤガヤ騒々しいのは，どこのどなたです？

Dr.D：影君のことかな？そういえば影君はどうしたのかな？

編集長：彼のことだから，またゲームをやっているんじゃないですか？

Dr.D：しょうがないな。どれマイコン・ルームにいってみるか。

編：そうしましょう。

（ドタ・バタ・ゴト・ゴト，イテッ……！とかなんとかいいつつ，舞台はマイコンルームへ！）

ガチャ（注：ドアをあける音です）。

影：あれっ，みなさん，もう集まってたん



ですか？いや～，申しわけないなあ(といいながら，ゲームに熱中している)。

Dr.D：仕事をさぼって何をやっているんだ！

編：あっ，あれはベーマガ'82年8月号で掲載したベーシックマスターJr.の「カバトリ・ゲーム」ですよ。

編集長：あれ以来，彼はよくカバトリ・ゲームをやっているようですよ。

Dr.D：影君にピッタリのゲームだな，最初の2文字をひっくり返えせば。

影：(ゲームに熱中していて，話が耳に入っていないようす) なんかいいました？

Dr.D：いや別に。影君，ゲームはそこでひと休みして，こっちへ来てくれたまえ。

影：いやですよ。

Dr.D：……。編君，この部屋はよく停電がおこるんだよね。

編：そ，そうですね。それも突然にね (といって，おもむろに電源を切る)。

影：あ～っ！消えちゃった。どうしよう。

〈リスト1－1〉カバトリ・ゲームのプログラム・リスト

```
10 LET C=$E890:POKE C+2,1:POKE C+1,6:DIM D(30),D1(30),H(501):LET G=0
15 LET N=20:REM ｶﾊﾞ ﾉ ｶｽﾞ *
20 LET S=0:LET O=-1:GOSUB 100
30 GOSUB 1000
40 GOSUB 230
50 FOR I1=0 TO I2:NEXT I1:IF O THEN GOTO 30
70 POKE C,$A5:LET CURSOR=5,10:PRINT "** GAME OVER **":MUSIC P3Uﾄｼﾗｿﾌﾐﾚﾄ:IF G<
S THEN LET G=S
80 LET CURSOR=26,18:PRINT "[M]":LET CURSOR=25,20:PRINT "[<][>]":LET CURSOR=25
,22:PRINT "[PUSH]";:IF INKEY$<>" " THEN GOTO 80
90 GOTO 20
100 POKE C,2:CLEAR :PRINT :FOR I=1 TO 32:PRINT "*";:NEXT I:FOR J=1 TO 21:PRINT
"*";SPC$(30);"*";:NEXT J
110 FOR I=$3E0 TO $3FF:POKE I,$2A:NEXT I:LET E=10:LET F=0:GOSUB 370
140 POKE C,4:FOR I=1 TO 25:LET M=INT(RND(19)+3)*32+INT(RND(26))+3+$100:POKE M,
$1A:NEXT I
150 POKE C,6:FOR I=1 TO 10:LET H(I)=I*$20+$290:POKE H(I),$95:NEXT I:POKE $2B0,
$94:LET B=1
155 LET D3=N:FOR I5=1 TO N:LET D1(I5)=$20
160 LET M=INT(RND(1)*22+1)*$20+INT(RND(1)*24+1)+$100
170 IF (PEEK(M)<>32)+(PEEK(M-3)<>32)+(PEEK(M-1)<>32)+(PEEK(M-2)<>32) THEN GOTO
160
190 IF PEEK(M+1)<>$20 THEN GOTO 160
220 LET D(I5)=M:LET I2=50:POKE C,5:POKE M,$9C:MUSIC T1P0Uﾄ:NEXT I5:POKE C,6:LE
T I5=1:RETURN
230 LET A$=INKEY$:IF A$="M" THEN LET F=-$20
240 IF A$=" " THEN LET F=$20
250 IF A$="," THEN LET F=-1
260 IF A$="." THEN LET F=+1
270 IF F=0 THEN RETURN
280 LET I=H(B):POKE I,$95:LET I=I+F:IF PEEK(I)=$9C THEN GOTO 340
290 IF PEEK(I)<>$20 THEN LET F=0:LET O=0
300 POKE I,$94:LET B=B-1:IF B<1 THEN LET B=500
310 LET H(B)=I:POKE H(E),$20:LET E=E-1:IF E<1 THEN LET E=500
320 IF I2>0 THEN LET I2=I2-1
330 RETURN
340 FOR I1=1 TO 3:MUSIC UｿDﾄUﾄDｿ:POKE I,$40:MUSIC ﾄﾚﾐ:POKE I,$14:NEXT I1:LET S
=S+10:LET E=E+1:GOSUB 370
345 FOR I4=1 TO N:IF PEEK(D(I4))=$14 THEN LET D(I4)=0
350 NEXT I4:LET I2=I2+4:LET D3=D3-1:IF D3=0 THEN MUSIC P4ﾐﾐﾌｿｿﾌﾐﾚ:LET S=S+100:
GOSUB 155
360 GOTO 300
370 POKE C,7:LET CURSOR=1,0:PRINT " GAME17 V.3 ﾊｲｽｺｱ";G;" ｽｺｱ";S:RETURN
1000 LET I5=I5+1:IF I5=N THEN LET I5=1
1005 IF D(I5)=0 THEN RETURN
1007 POKE C,5:LET D(I5)=D(I5)-1:LET D1(I5)=PEEK(D(I5))
1010 IF D1(I5)=$2A THEN POKE D(I5)+1,D1(I5):LET D(I5)=D(I5)+30
1020 POKE D(I5),$9C,D1(I5):IF D1(I5)=$95 THEN POKE C,$26:POKE D(I5),$2A,$2A:LET
O=0
1030 POKE C,6:RETURN
```

編集長：子供みたいに，いつまでもダダをこねてるからだ。

影：エーン！エーン！

Dr.D：調子にのるな！

編：ところで，誌面も限られてますし，そろそろ本題に入りますか。

Dr.D：本題って何だい!?

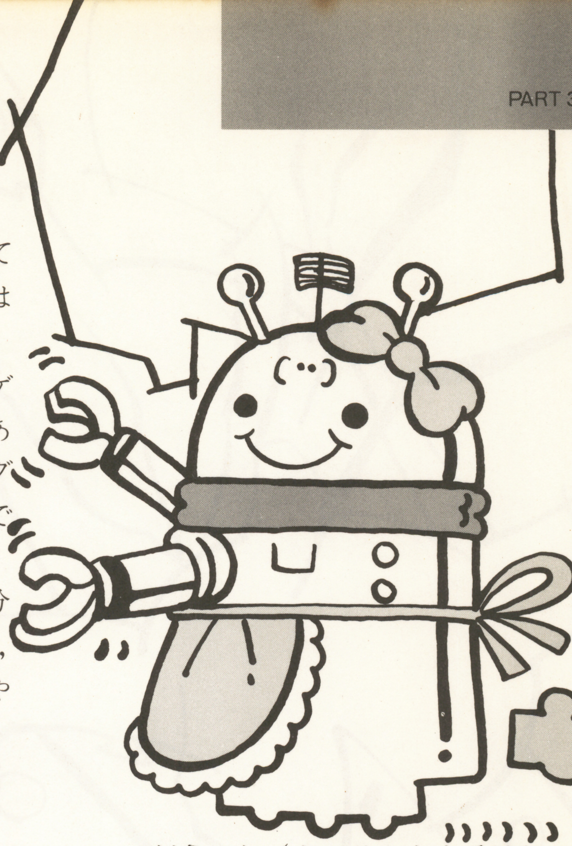
影：本題って本だい！

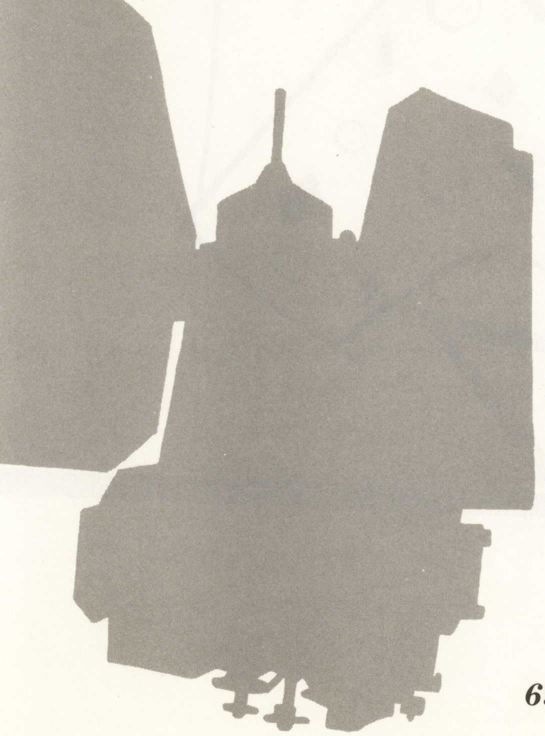
編集長：じょうだんをいっている場合じゃない，今日はいそがしいんだから！

編：せっかく，カバトリ・ゲームのリストが手もとにあるんですから，このプログラムを分析してほしいんですけど……。

Dr.D：ほ〜っ，プログラム分析か，久しぶりだな。よし，ちょっと気合いを入れてやってみるか！

編：それじゃ，ガンバッテください ね（と，リストを手渡して立去ろうとする）。





Dr.D：オイ！オイ！ワシをおいて，まさか帰る気じゃあるまいな……！

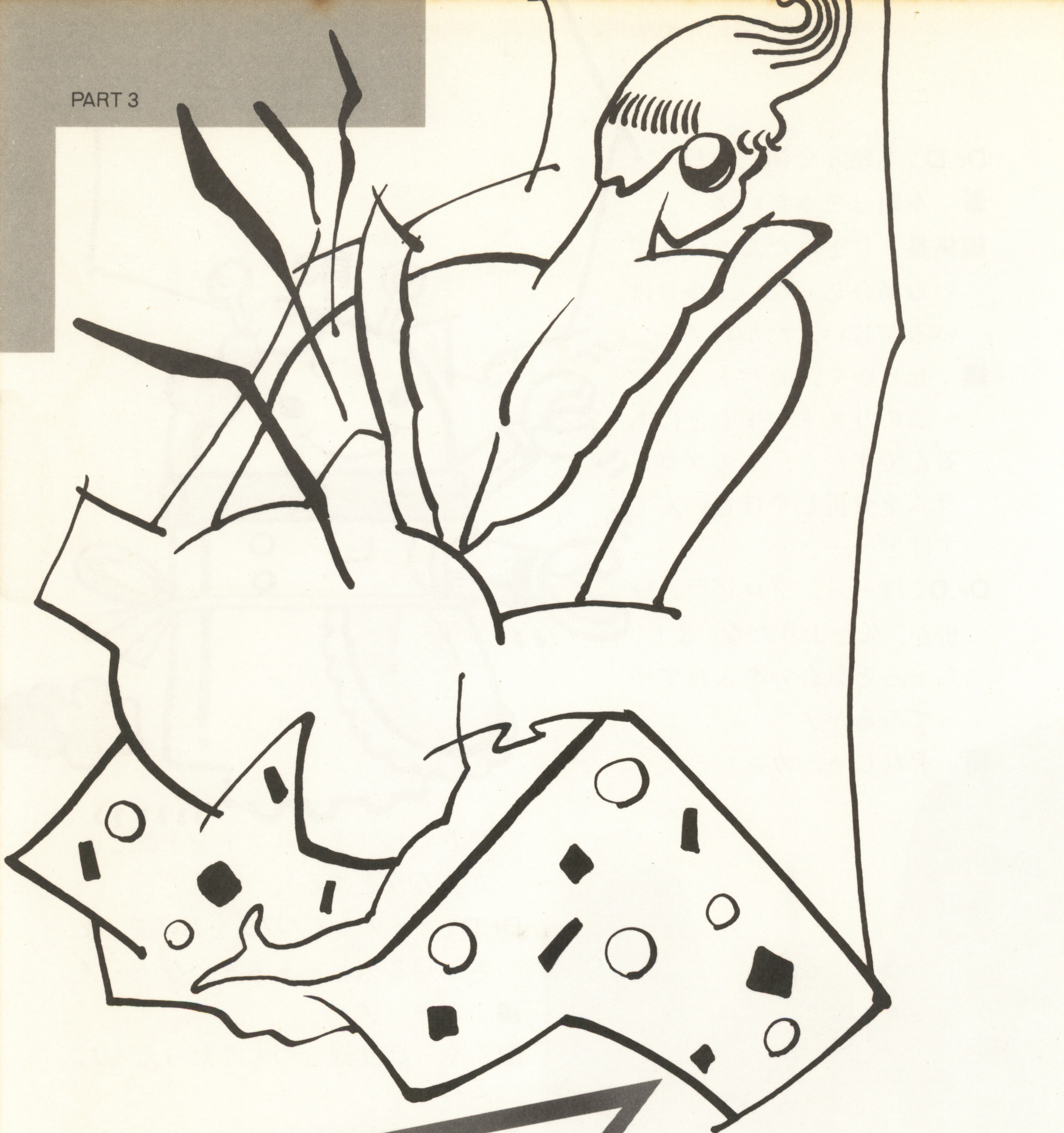
編：ドキッ！

影：やっぱり帰るつもりだったんだ〜!?

Dr.D：影君も帰ろうとしていたな（と，冷やかな目で）。

影：ギクッ！

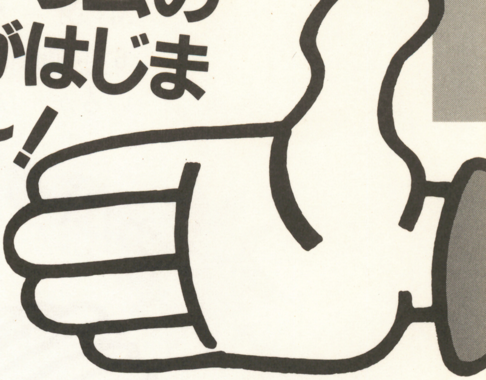
編集長：まあ許してやってください。2人とも帰るなんていうのは失礼だぞ。仕事を依頼したんだろ(実は編集長も帰るつもりだったんだって：**影**)。



それでは，私達も手伝いますから，プログラムを分析していってください。

Dr.D：よし，それではいくぞ!!

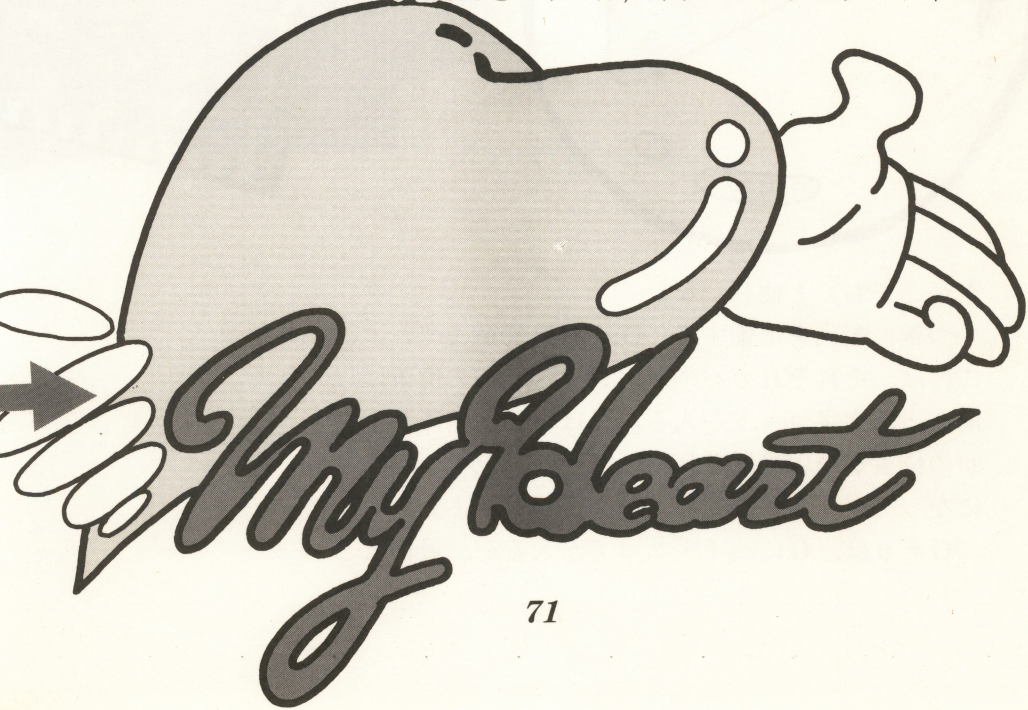
プログラムの分析がはじまるよ〜！



Dr.D：まず手はじめに10行目だ。ここは影君に説明してもらおうか。

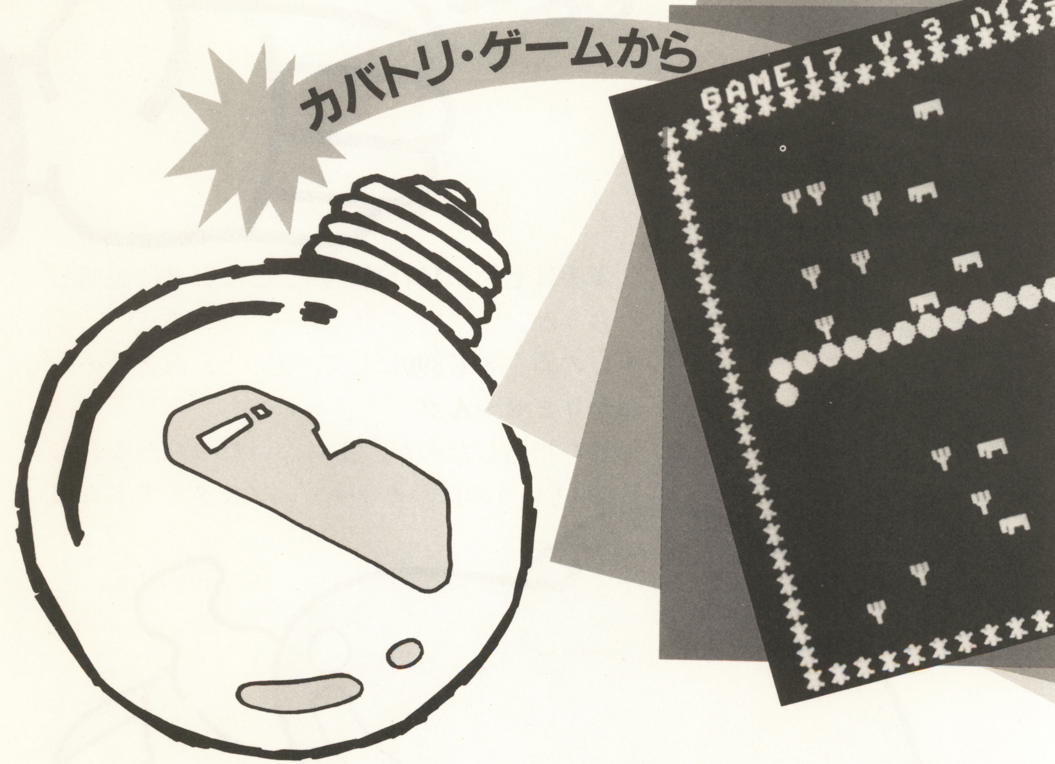
影：まずCの値を＄E890にして，え〜と，あとは…………**わかりませ〜ん!!**

Dr.D：影君を指名したのが，まちがいだったかな。C＝＄E890というのは，カラー・レジスタ・アドレス



の＄Ｅ890をＣに置きかえるという意味だ。

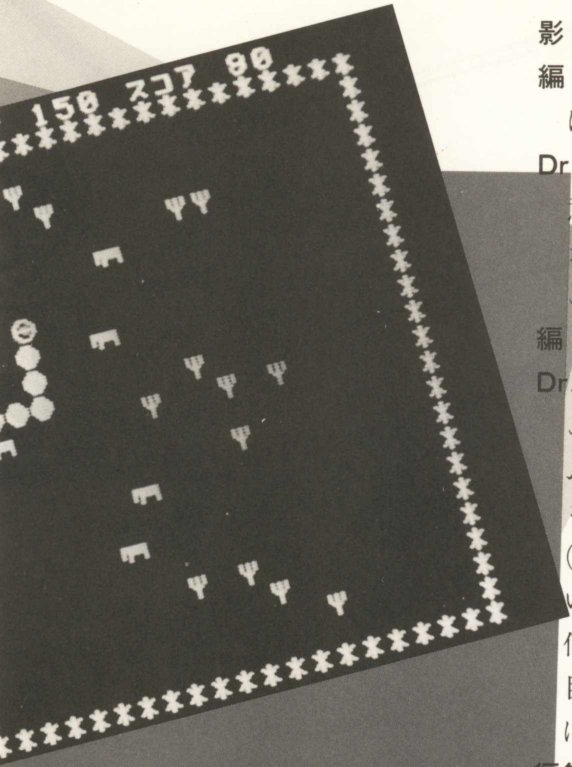
10行目のＰＯＫＥＣ＋２，１（これはカラーＲＡＭのオープンを指示している）や，ＰＯＫＥＣ＋1，6（画面背景の色を黄色にする）でこのＣを使っているが，ひんぱんに使うものはこのように変数に置



きかえておくと便利だな。

つぎに配列を準備する。Ｄ，Ｄ1は30個までだ。ここには，それぞれカバのアドレス，カバの位置に以前あった文字コードが入る。Ｈはヘビの胴体および頭の位置が順次格納されていく，サイズがなぜ501個になっているのかは，考えてみてくれたまえ。

Ｇ＝０は，Ｇにハイ・スコアが入るが，最初はこ



れを０点にしておく。

わかったかね，影君!!

影：ブツブツ…！

編：15行はわかりますよ!!カバを20匹にするんでしょ！

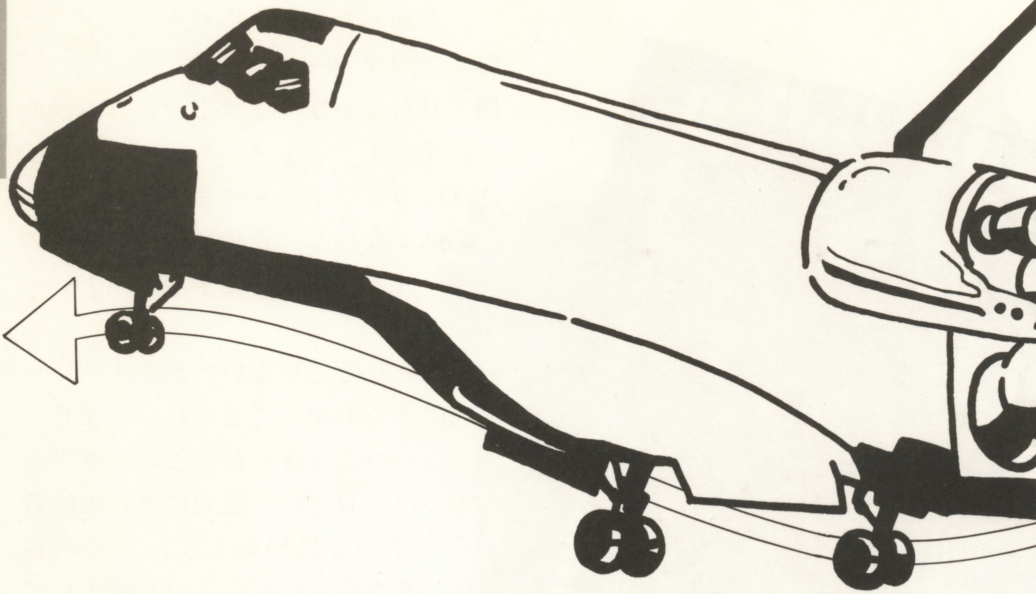
Dr.D：コメントをそのまま読んじゃ，意味がないな。正確には，何番目のカバまであることにするか…ということだ。

編：同じことじゃないのかな？

Dr.D：よくわかってないね。つまり，こういうことなんだ。このプログラムでは，Ｄという配列にカバの位置をつぎつぎに格納して，その内容(値)を参照しながらカバを動かしている。つまり，Ｎの内容というのは，何匹いるかということよりも，何番目の配列まで参照するかということになるわけだ。

編集長：むずかしいんですね。わからなくなりそうですよ。

Dr.D：まあ，そんなこといわず，つぎにいこうか。20行は，Ｓ（スコアが入る）を０にする。また，Ｏ（ゲーム・オーバー・スイッチ）を－１にする。この変数Ｏは，ゲーム・オーバーのとき以外は－１になっていて，ゲーム・オーバーのときは，０になる。さらに，ＧＯＳＵＢ100で100行からのサブルーチンにとんでいく。



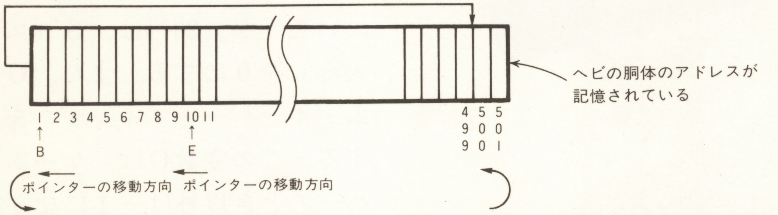
100行からは，画面作成サブルーチンだ。

影：ＺＺＺ……！

編：あれ，影さんたらいねむりしてるよ。

編集長：そのほうが静かでいいから，放っておこう。

■第１－１図　配列Hの使われかた



これは，First in First out つまり，Bの方向から押しこまれて，Eの方向から，出てくるようなイメージだ。
また配列Hの状態は，図のようにH(500)とH(1)がつながってしまうように使われる。
H(501)は，E(ヘビ胴末ポインタ)が500になっているときに，カバを食べ，胴をのばすときのE＝E＋1に対応するためにある。



Dr.D：話を続けるぞ。30行，40行はサブルーチンに飛ばしている。ＧＯＳＵＢ　1000の1000行はカバ移動のサブルーチンだね。

また40行のＧＯＳＵＢ 230は，ヘビを動かすサブルーチンだ。**わかったかい，影君!!**

影：ビクッ!!

Dr.D：やっと目が覚めたか。しっかり聞かなきゃだめだろ。

編：Dr.D, 影さんが眠くなるのも当然のような気がするんですよ。

編集長：そうですね，どうも言葉と文字(リスト)だけじゃ，分りづらいんですね。Dr.の説明がくわしければくわしいほど，わかりづらくなりますね。

影：そうなんです。もう頭の中が暴走しているんですよね。

Dr.D：ほほう，頭が暴走するほどまでに影君はまじめに聞いていてくれたのか？

影：ガクッ！

編：なにかパッと見て，プログラムの流れがわかるものはないんですか？

Dr.D：君たちがそういうのを待っていたんだよ。

今回は「プログラム分析」ときまって，すぐにいおうと思っていたのだが，自分たちで気づくまではだまっていようと思ってね。

編集長：なにかいい方法でもあるのですか？意地悪しないで教えてくださいよ！

Dr.D：これをみればプログラムの流れがよくわかるという図がある。図の名称はフローチャートという。

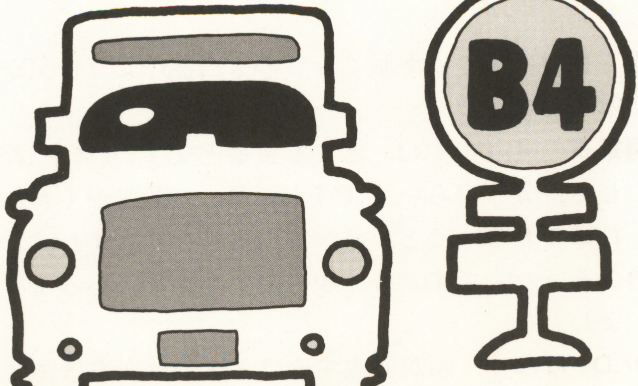
影：風呂をちょっとて何ですか？

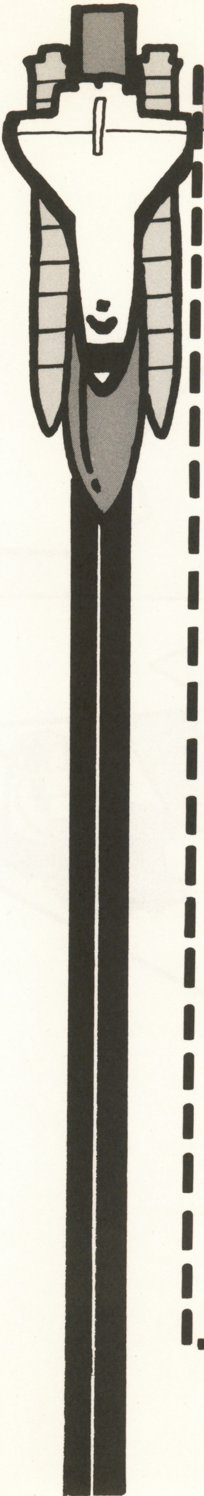
編：フローチャートですよ。流れ図ともいいます。

影：それで，どうやって書くんですか？

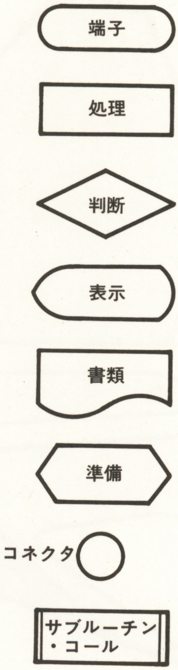
Dr.D：まず書きはじめる前に，約束ごとを覚えてほしい。

第1－2図を見たまえ。フローチャートではこんな記号を用いるんだ。矢印の書きかたにも規則があるんだが今回はそのアウトラインだけにとどめておこう。





■第1－2図　フローチャートで用いる記号



端子：始めや終わりを示す

処理：一般的な作業(計算など)を示す
一連の作業をまとめて書くときなどにも用いる

判断：判断文を示す。FOR～NEXTもこれで記述する
一般に，IF文やON～GOTO(GOSUB)など

表示：画面への表示。V-RAMへのPOKE文なども含まれる

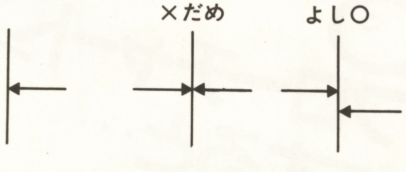
書類：プリンタなど記録が残るものへの表示

準備：配列の宣言など，プログラム開始時の準備を示す

コネクタ：フローチャートの線で離れた部分などにつなぎたいときに使われる

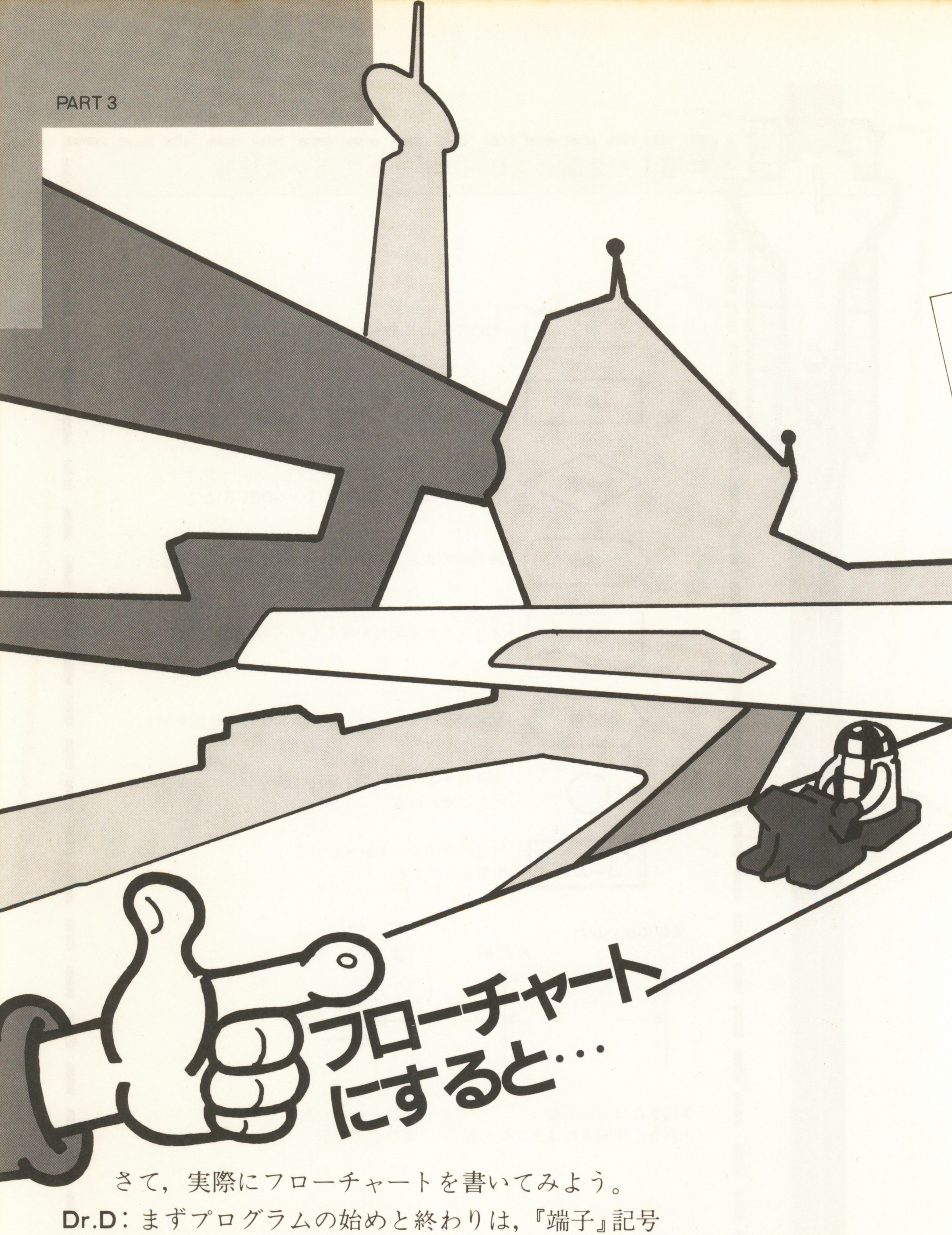
サブルーチン・コール：指定のサブルーチンへ行って作業をしてくる

矢印の使いかた



下向きには矢印不要
上向き，横向きには矢印が必要

矢印が接するときは
ズラして表示

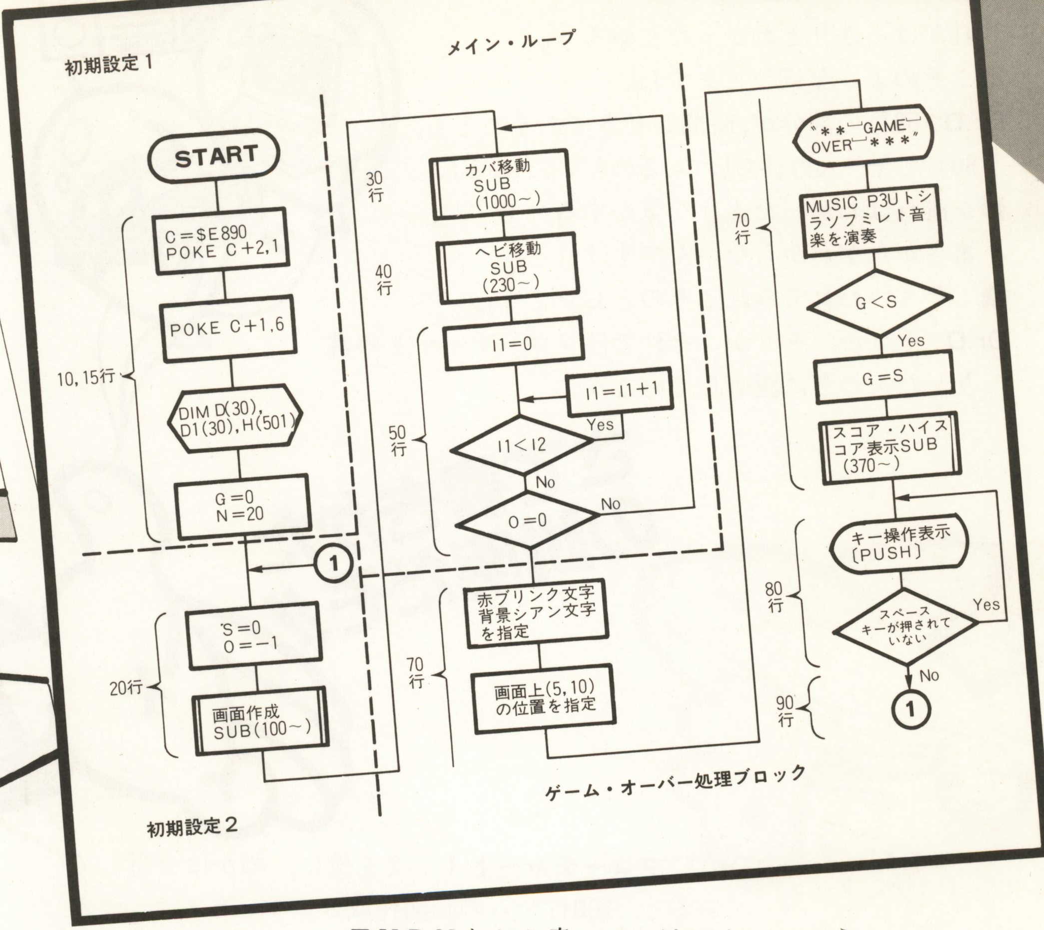


フローチャートにすると…

さて，実際にフローチャートを書いてみよう。

Dr.D：まずプログラムの始めと終わりは，『端子』記号を使う。なかにはSTART，はじめ，終わり，RE

▼フローチャート1－1(メインルーチン)

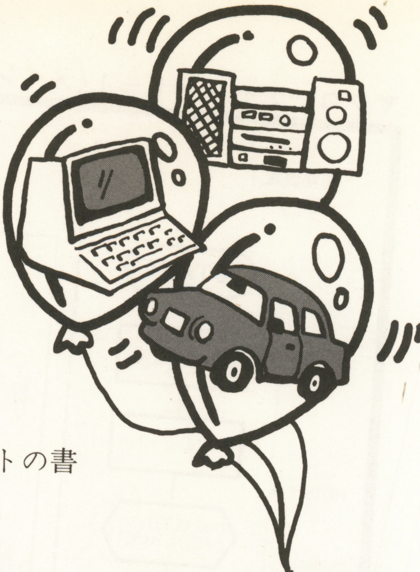


TURNなどと書いておけばよいだろう。

GOTO文は，とんでいく行の手前までを，矢印で結ぶ。線をひくのがたいへんなときは，**フローチャート1－1**のように，『コネクタ』を使って（①となっているもの）連結してもかまわない。

このようにして書いたのが，**フローチャート1－1**のメイン・ルーチンだ。FOR～NEXT文はIF文式の記述になおした。

しかし，このようにみると，メインルーチンの流



れがはっきりとわかったじゃろう。
編：そのような気がしますね。
Dr.D：50行の部分が，時間かせぎであることも，80行でキー入力待ちしているのもすぐわかるね。
影：流れはわかったような気がするのですが，書きかたがわからないんですけれど。
編：そういわれてみればそのとおりですね。
Dr.D：そうか，そうか。それではフローチャートの書きかたをつぎに説明していこう。

さて書きかたの説明だぞ!!



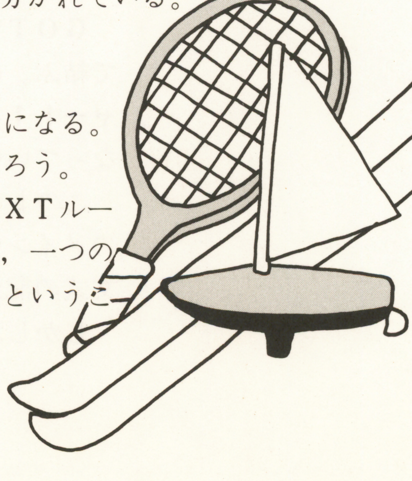


Dr.D：**フローチャート1－2**を例に，細かに分析してみると，100行からの画面作成サブルーチンは，つぎの二つのブロックに分かれている。

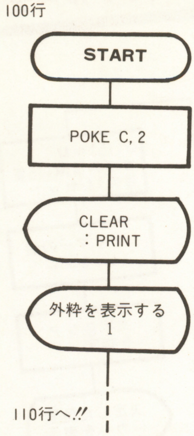
1．枠表示部
2．カバ表示部

まず100行だ。100行は**第1－3図**のようになる。実際の行と対比してみるとわかりやすいだろう。

外枠の表示部は，ここではＦＯＲ～ＮＥＸＴループを使っているから，くわしく書くならば，一つの記号にはまとまらないが，内容がわかればということで一つにまとめている。



■第1－3図　100行のサブルーチン分析



サブルーチンのはじめだから
スタートを書く

POKE C, 2 は
カラー指定だから処理でよい

クリアで画面をけして１行さげる。
つまり画面操作だからこの記号！

複数行をまとめて，一つの記号で記述する。
FOR I＝1 TO 32：PRINT "＊"；：NEXT I：
FOR J＝1 TO 21：PRINT "＊"；SPC$(30)；"＊"；：NEXT J

フローチャートでは，外枠を表示する部分を一つにまとめて書いてあるが，ここでは行単位に書くため，二つに分けて書くことにする.!!(Dr.D)

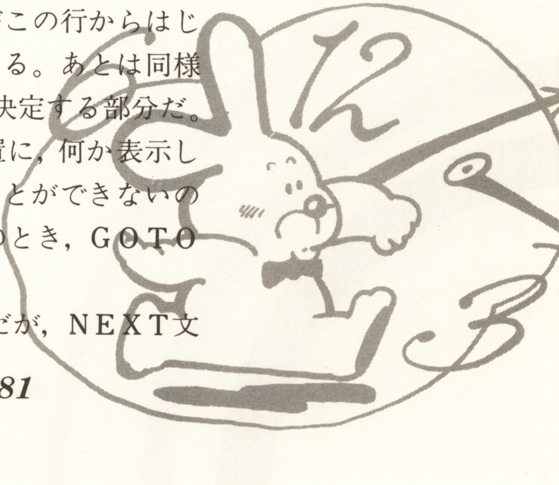
110行の部分は，前の２ステップは100行と同じになる。そして，最後のGOSUB 370は，**第１－４図**のような記号で書く。

140行だが，サボテンを表示するという部分も一つの記号にまとめる。150行も同様だ。

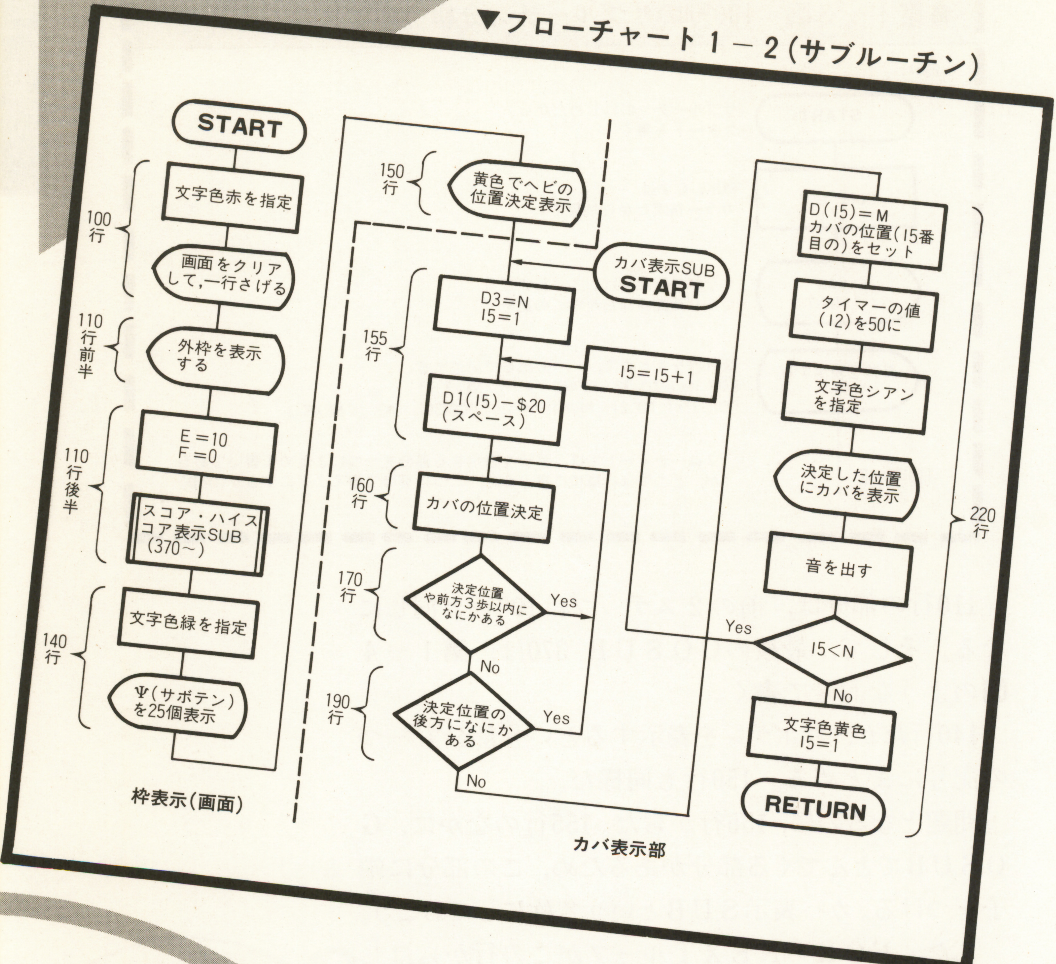
問題となるのは，155行からだ。155行のなかに，GOSUBでとんでくる部分があるため，この部分に端子をつける。カバ表示SUBという名称にしておこう。

また，FOR～NEXTループがこの行からはじまっているので，ループの処理をとる。あとは同様だね。160行はカバの位置を乱数で決定する部分だ。170，190行は，160行で決定した位置に，何か表示してあったときは，カバを表示することができないので，160行で位置を決めなおす。このとき，GOTO文は矢印で表示するわけだ。

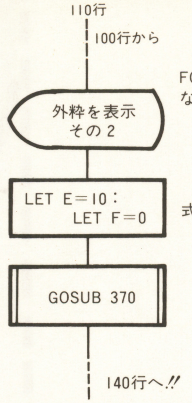
220行は，以前のステップと同じだが，NEXT文



▼フローチャート1－2(サブルーチン)



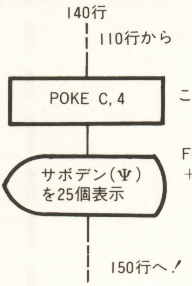
■第1－4図　110行のサブルーチン分析



FOR I=$3E0 TO $3FF：POKE I,$2A：NEXT I
なぜPOKEでやったか？スクロールさせないため!!

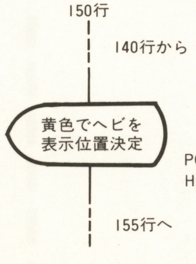
式だから，処理でよい

■第1－5図　140～150行のサブルーチン分析



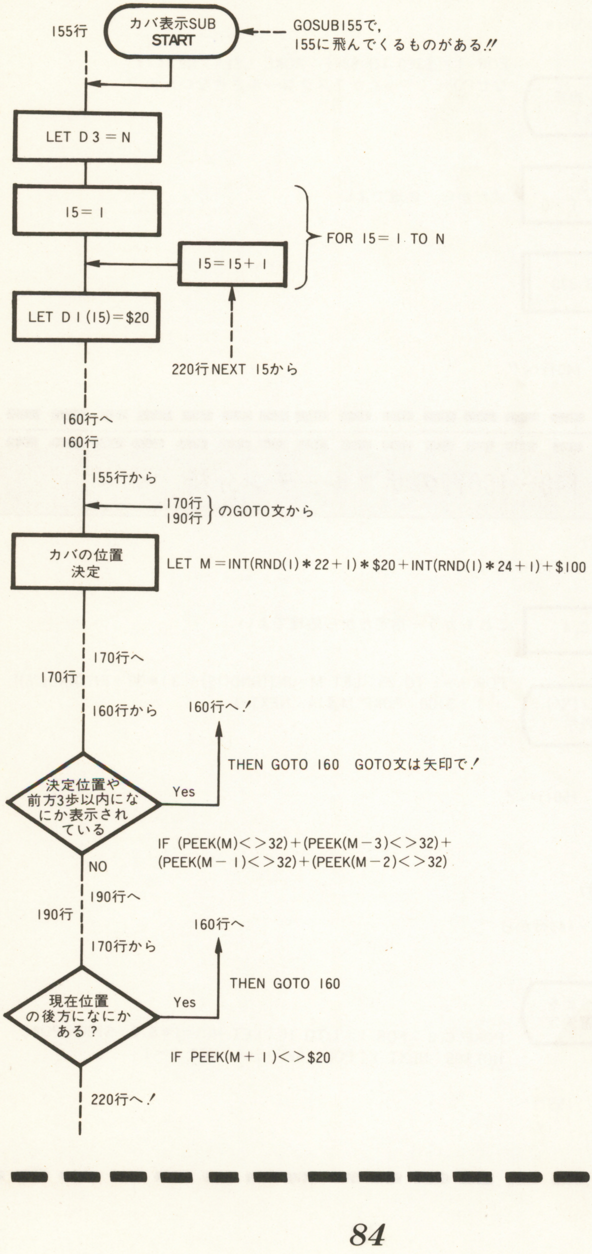
これもカラー指定だから処理でよい

FOR I=1 TO 25：LET M=INT(RND(19)+3)＊32+INT(RND(26))+3+$100：POKE M,$1A：NEXT I



POKE C,6：FOR I=1 TO 10：LET H(I)=I＊$20+$290：POKE H(I),$95：NEXT I：POKE $2B0,$94：LET B=1

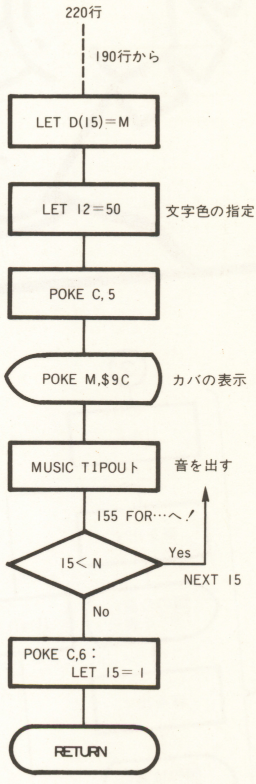
■第１－６図　155～190行のサブルーチン

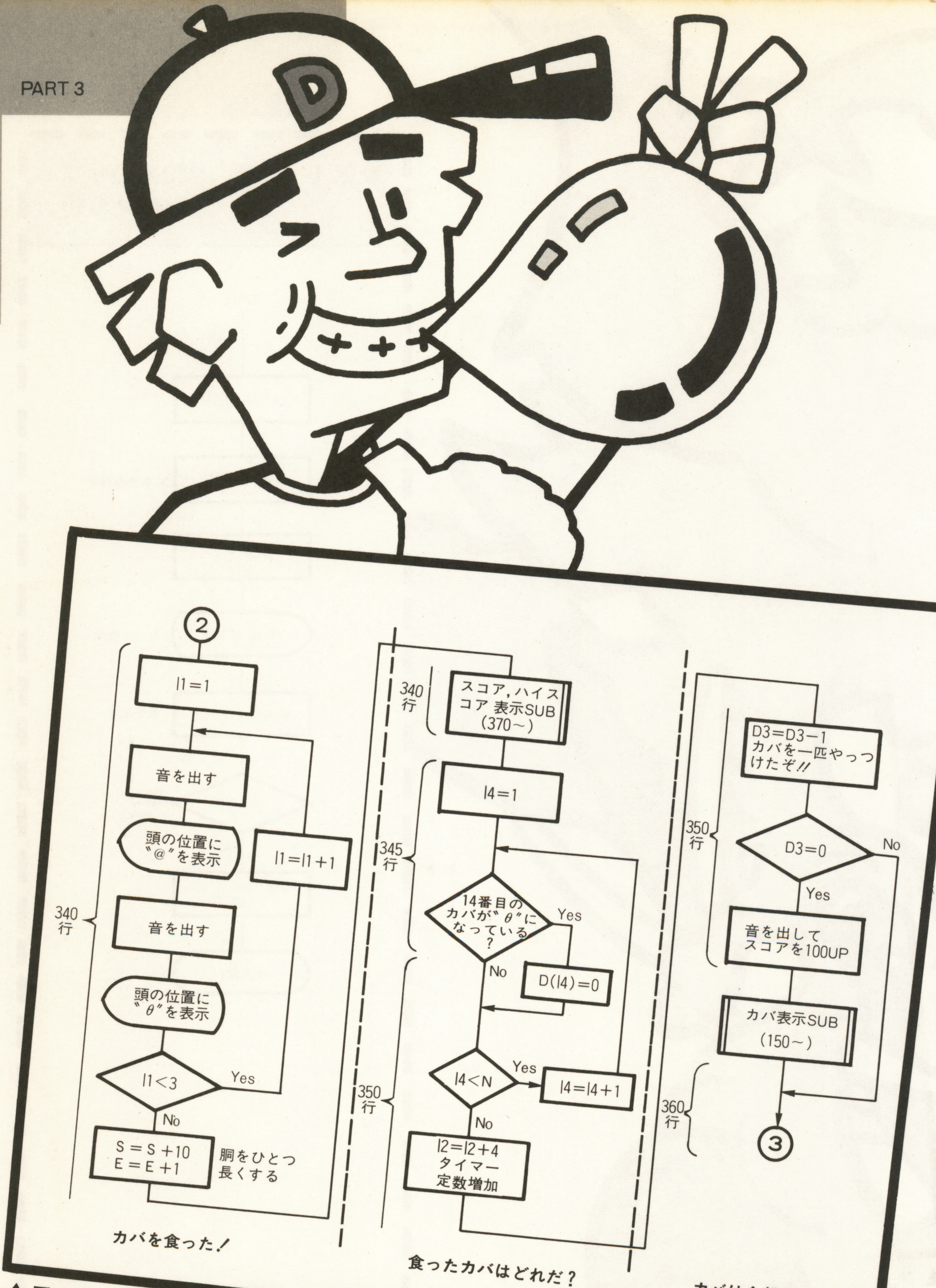




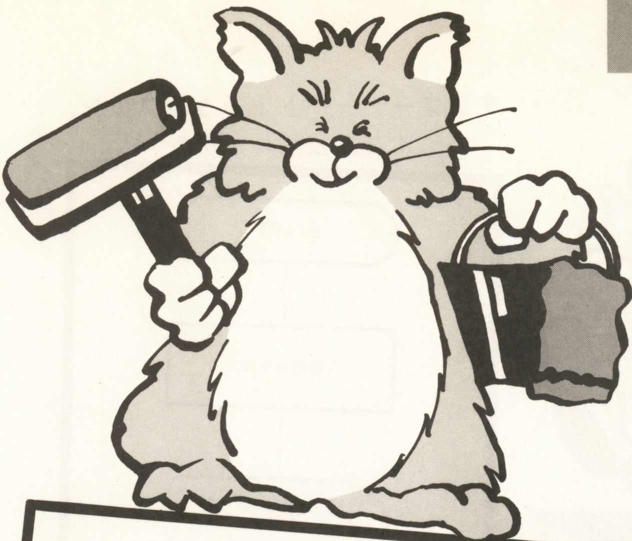


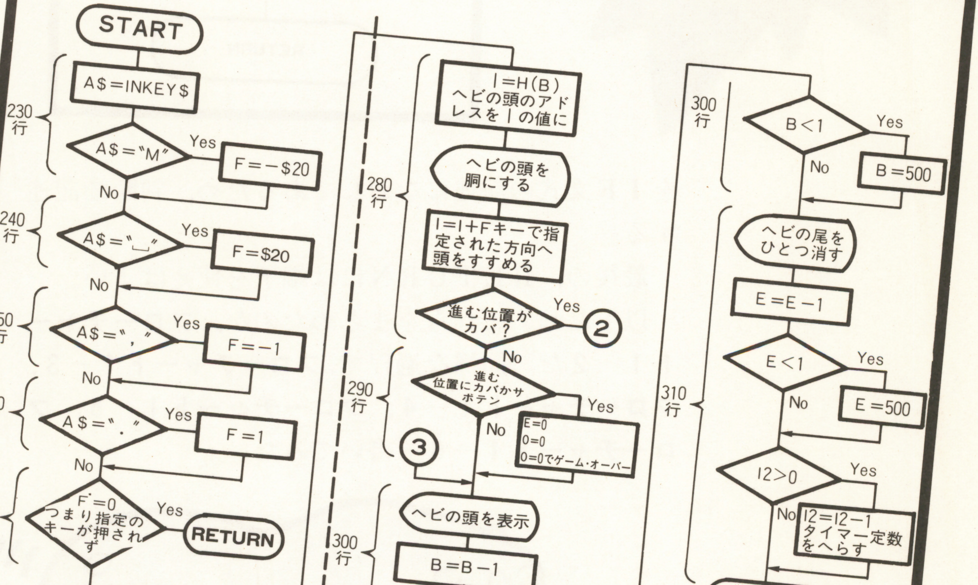
■第1－7図　220行のサブルーチン分析





▲フローチャート1－3（ヘビ移動のサブルーチン）

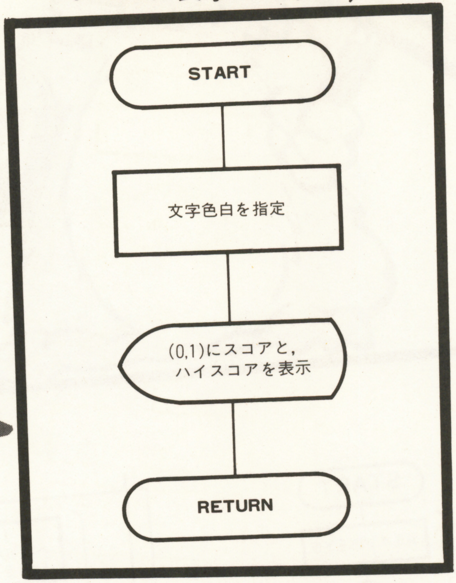




▲フローチャート1−4(ヘビ移動のサブルーチン)



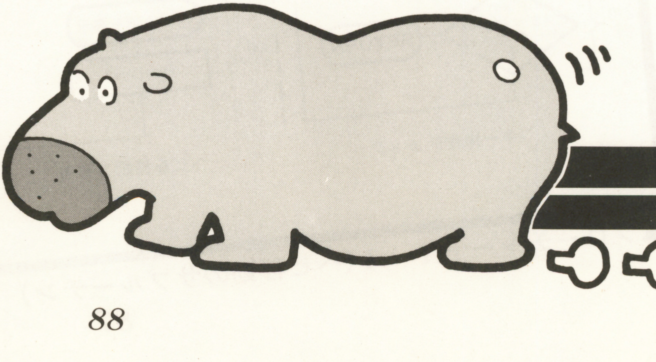
▼**フローチャート１－５（スコア，ハイスコア表示ルーチン）**



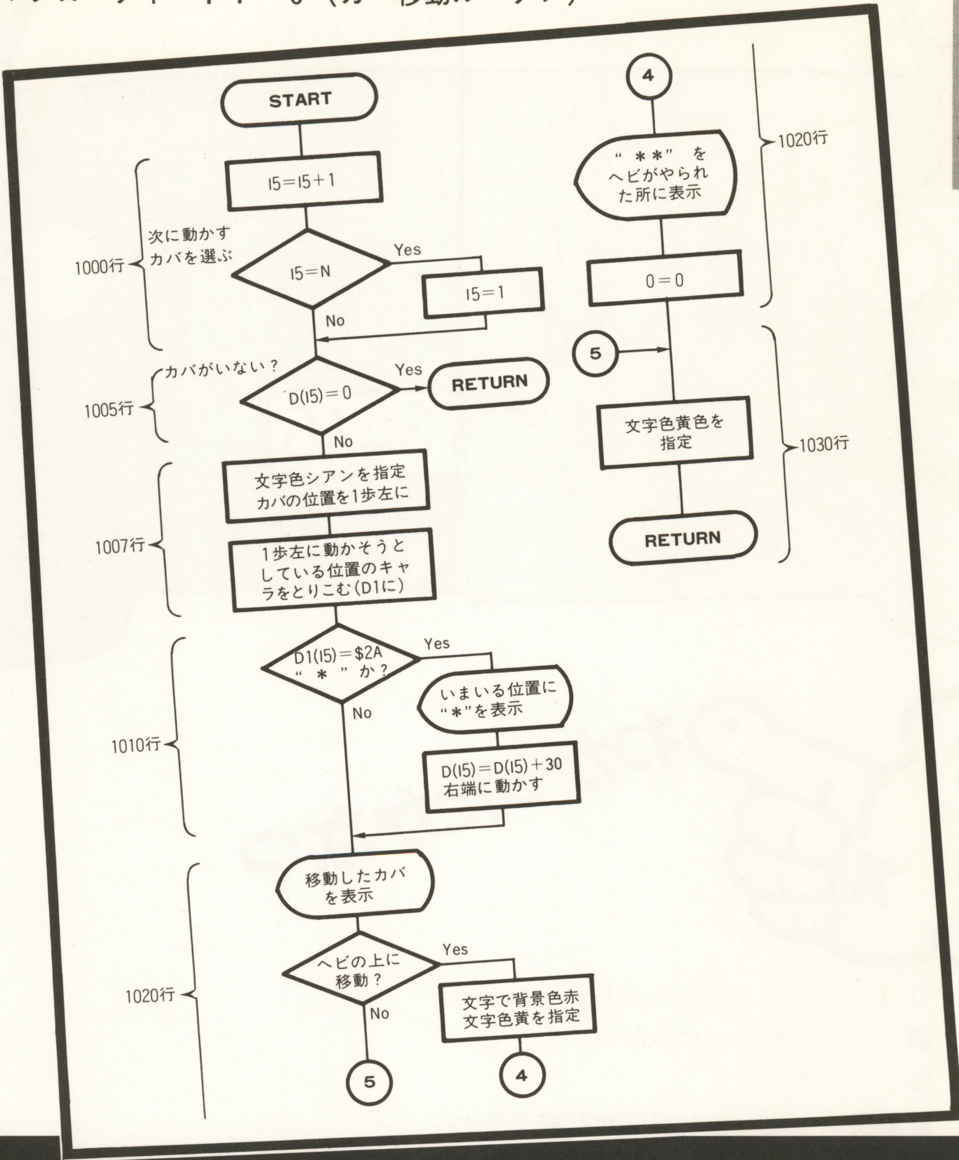
をＩＦ文式の記述になおしてあるため，判断で記述する。

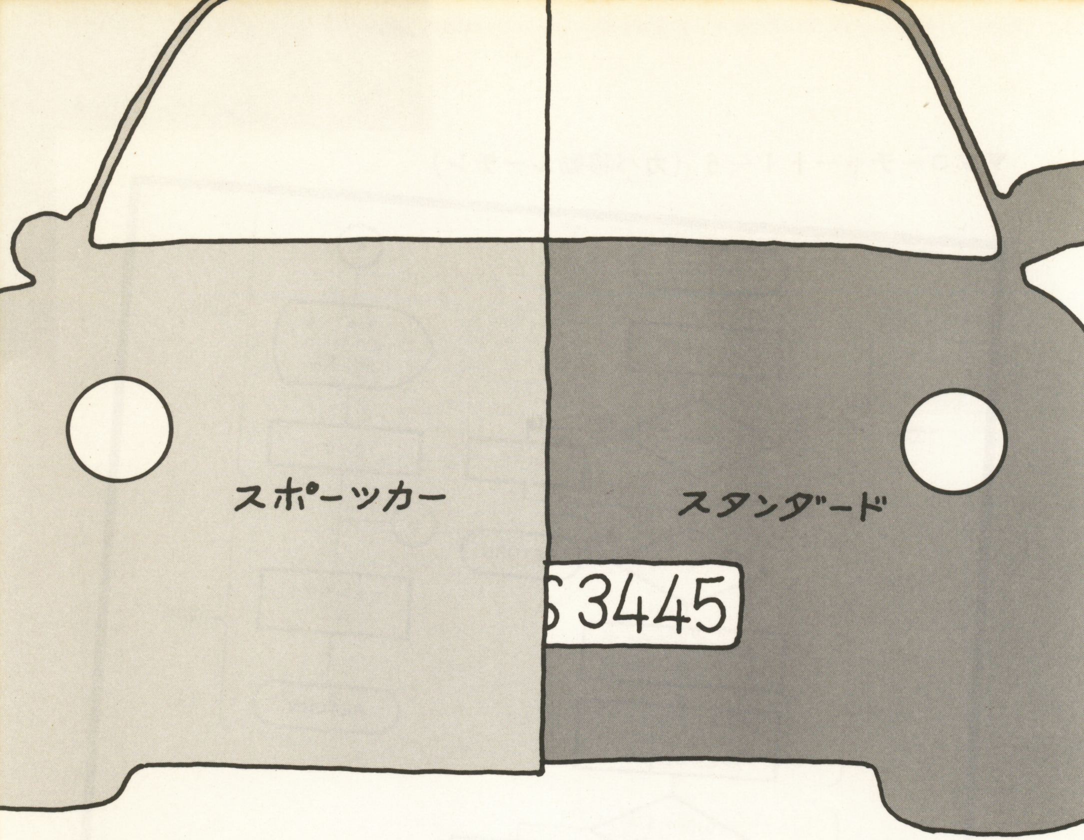
最後の，ＲＥＴＵＲＮには端子を使えばよい。

以上だね。これらをまとめたのが，**フローチャート１－２**だ。同様な過程で，**フローチャート１－３**，**フローチャート１－４**，**フローチャート１－５**，**フローチャート１－６**を書いてみた。



▼フローチャート１－６（カバ移動ルーチン）







Dr.D：これで，君たちもわかったかな？
編：やっとわかったみたいです。
影：バッチリ！
編集長：どうもまだ自信がないですね。
Dr.D：そう，実際に自分で書いてみることが一番の近

道だし，プログラムの内容を知るためのよい勉強にもなるな。

このようにフローチャートを書くと，プログラムの流れがわかり，ムダな部分や簡単にできる部分をみつけることができるわけだ。

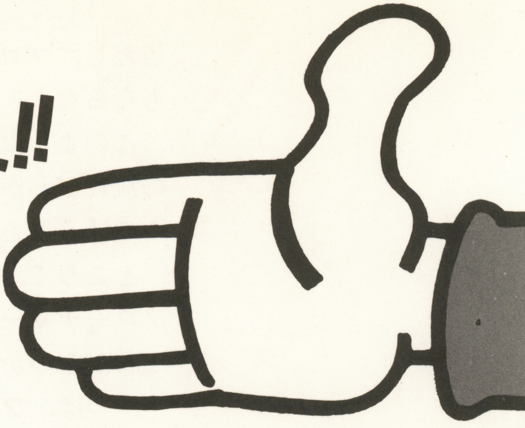
ところで，このカバトリ・ゲームのプログラムは，フローチャートを書いてみたら，2か所ほど気になる所がみつかったぞ。

一つは，**フローチャート1－6**のカバ移動ルーチン内のI5＝Nの条件分岐の部分だ。ここは，I5＝N＋1としないと，N番目のカバが動かないね。ゲームでは，動かないカバが一匹ぐらいいてもさしさわりないが，明らかにミスだね。

またD1が配列になっているが，このプログラムのかぎりでは，D1は配列にする必要性はない。なぜならばつぎの位置の文字をとりこんでも，つぎのステップで使われることがないため，一匹一匹のカバごとにとっておく必要がないためだ。



まとめだよ!!



Dr.D：では，まとめてみよう。

1．プログラムを作る前はフローチャートを書いて，

■第1－16表　変数の役割

変数	役割
C	カラーレジスタのアドレス（$E890）が入っている
D	カバの位置が入る（アドレスで）　配列（DIM）
D1	カバの位置に以前書いてあった文字のコードが入る　配列（DIM）
H	ヘビのアドレスが入る
G	ハイスコアが入る
N	カバの数（配列の数を考えると30個まで増やせる）
S	スコアが入る
O	ゲーム・オーバー・スイッチ（0が0ならゲームオーバー）
I, J, M, I4	WORKING変数
F	キー操作により進める方向定数が入る
A$	押されていたキー
B	ヘビの頭の位置を示す，配列Hのインデックス変数（第1－1図参照）
E	ヘビの最後の位置を示す，配列Hのインデックス変数（第1－1図参照）
I2	タイマー定数。I2の値が小さくなるほど早くなる
I5	カバの位置を示す，配列，D，D1のインデックス変数
D3	あと何匹のカバがいるか？

全体的な流れをきめよう。そうすれば，GOTO文のスパゲッティはなくなる。

2．他人の作ったプログラムの内容分析にも，フローチャートが役立つ。内容分析が完璧ならば，プログラムの移植もたやすい!!　以上



影：Dr.がえらそうに話をまとめちゃったけど，編君わかった？

編：んー，その一，まあ，それ，なんというかー。

Dr.D：ブッブッ何をいっているんだ。

編：ハ，ハイ！十分よくわかりました（**編集長**：編君は，本当にわかっているようだが，影君はまったくわかっていないね!!）

編集長：それでは，わからなかったのは，影君だけだったようだから，彼は無視しておわりにしましょう。

編：みなさん，影さんにならないように，しっかり覚えましょう。



メモリを直接アクセスする命令PEEK

編：博士！（返事がない）博士いないんですか？

編集長：博士は，カゼを引いて寝込んでいるらしい。先ほど連絡があった。編君，質問があるのなら私に聞きなさい。

影：編集長，質問なんかに答えられるんですか？

編集長：だてに年を取ったわけじゃないんだぞ。

影：マイコンの世界では，年の功は通用しないと思うんですが……。

編：まあ，何はともあれ，質問をしてみます。

読者から送られてきたプログラムの中に，〝PEEK〟とか〝POKE〟とかいう命令が出てきたんですが，なんだかさっぱりわからないので教えてほしいんですが。

編集長：ジーコン，ジーコンと！男は黙って電話のダイヤルを回す！（ブッブッと一人言を言う）

影：思った通り，神頼みが始まった。

編：編集長！教えてください。

編集長：（あわてて席を離れ，トイレにかけ込む）

　―――１時間近くたち

影：編集長，なかなかトイレから出てきませんね。もう１時間もたつというのに。

　（コン，コン，ガチャ！ドアが開いて，人が入ってくる）

編：あれっ！丹治先生ではありませんか？パソコンの初期設定では，おせわになりまして。

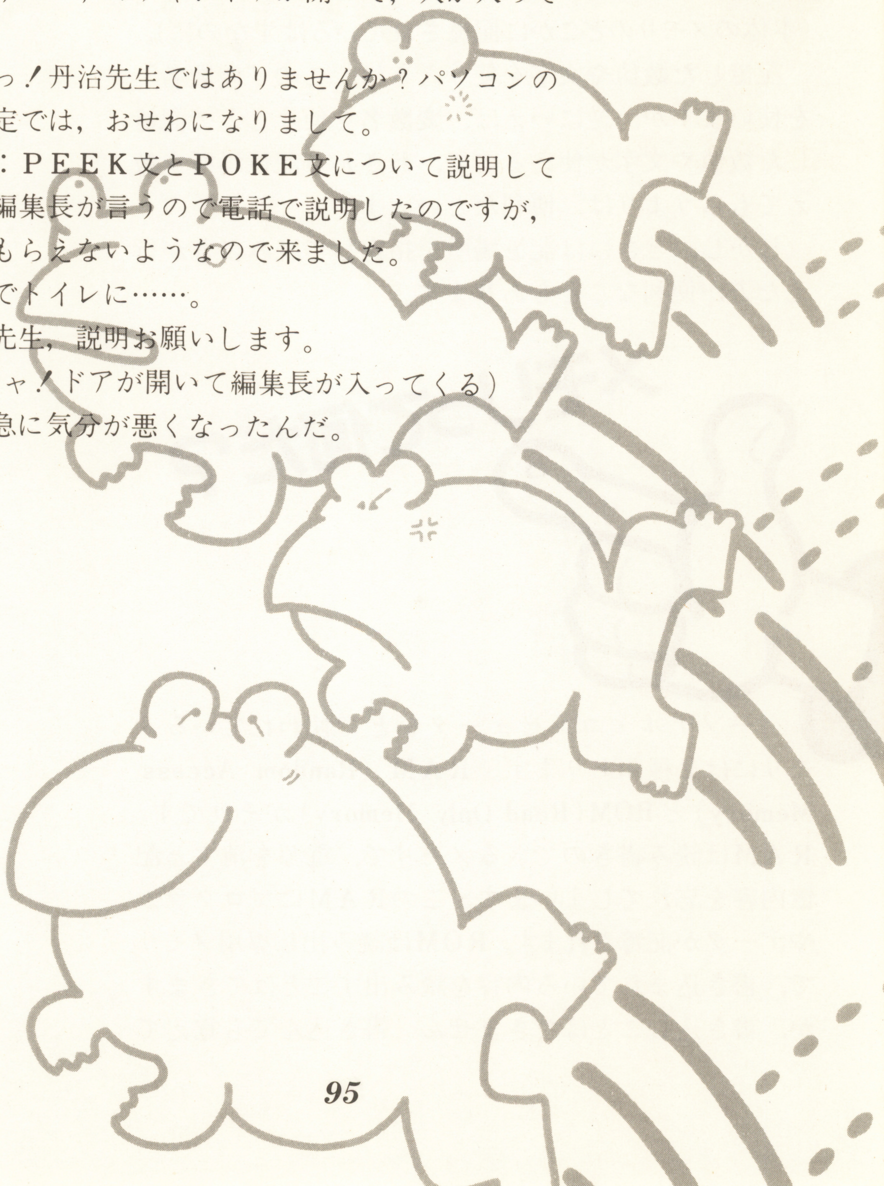
丹治先生：ＰＥＥＫ文とＰＯＫＥ文について説明してくれと編集長が言うので電話で説明したのですが，分ってもらえないようなので来ました。

影：それでトイレに……。

編：丹治先生，説明お願いします。

　（ガチャ！ドアが開いて編集長が入ってくる）

編集長：急に気分が悪くなったんだ。



　適当に見つくろって説明してください。
丹治先生：それでは，説明を始めます。

　いままで勉強してきた中で，メモリを使う命令に変数がありました。この変数は，数値や文字を記憶できるものですが，記憶場所は私たちに教えてくれません（本体のメモリのどこかに記憶されているはずなのに）。

　記憶した数値や文字を使うときは，この“変数名”を使いますが，逆にいえば，変数名を使うことで記憶した数値や文字が使えるので，わざわざ記憶場所を教えてもらうよりは，便利だといえます。

　しかし，ときには記憶場所を指定して読み書きが行えた方が便利なこともあります。



　パーソナル・コンピュータなどで使われているメモリには2種類あります。ＲＡＭ（Random Access Memory）とROM（Read Only Memory）がそれです。ＲＡＭは読み書きのできるメモリで，電源を消すと記憶内容を忘れてしまいます。このＲＡＭにプログラムやデータが記憶されます。ROMは読み出し専用メモリで，書き込まれている内容を読み出すことはできますが，書き込むことはできません（書き込んでも覚えて



くれません)。ROMは電源を切っても内容が消えることがありませんから，BASICのインタプリタなどを書き込むのに使われています。また最近では，パソコンで漢字を出すために，漢字の形をROMに覚え込ませています。

これらメモリには，記憶場所を示す住所(アドレス)が付けられ，読むにしても書くにしてもすべて“アドレス”を指定して行います。

8ビットのCPU(マイコン)を使用したパソコンでは，一つのデータは8ビットで表されますから，0から8ビットで表すことが出来る最大値255(2^8-1)までを一つのアドレスに記憶することができます。同様に，アドレスは16ビットで表されますから65536(2^{16})個のアドレスが存在します。



メモリの内容を読み出す命令 PEEK

メモリのあるアドレス（番地）を指定して，その内容を読み出す命令です。これは，変数に書き込まれた

内容を見ることと同じです。

命令は“PEEK”文で，ピークと読みます。

さてその使い方ですが，次のようになります。

メモリ・アドレスの内容が代入される

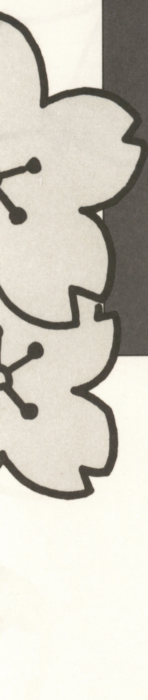
必ず（　）を付けること

変数 ＝PEEK（**アドレス**）

ピーク文（PEEK）

メモリの番地（アドレス）

一つのメモリ番地(アドレス)に書き込まれている値は0～255です。よって変数に代入される値は，0～255までの10進値になります。



アドレスの表しかた

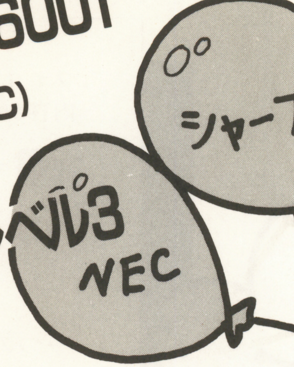


アドレスは，10進値で表す機種と16進値で表す機種の2種類があります。これは，アドレス(メモリ番地)指定の方法に限らず16進値を表すための方法でもあります。

アドレス（メモリ番地）は10進値で0～65535，16

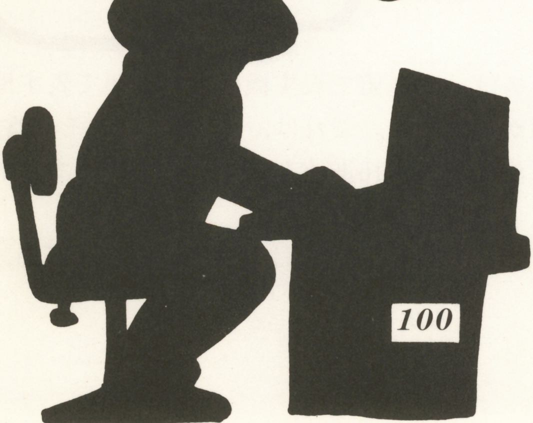
進値では０～ＦＦＦＦということになります。また８ビットで表すことができるデータの値は，10進値で０～255，16進では０～ＦＦとなります。

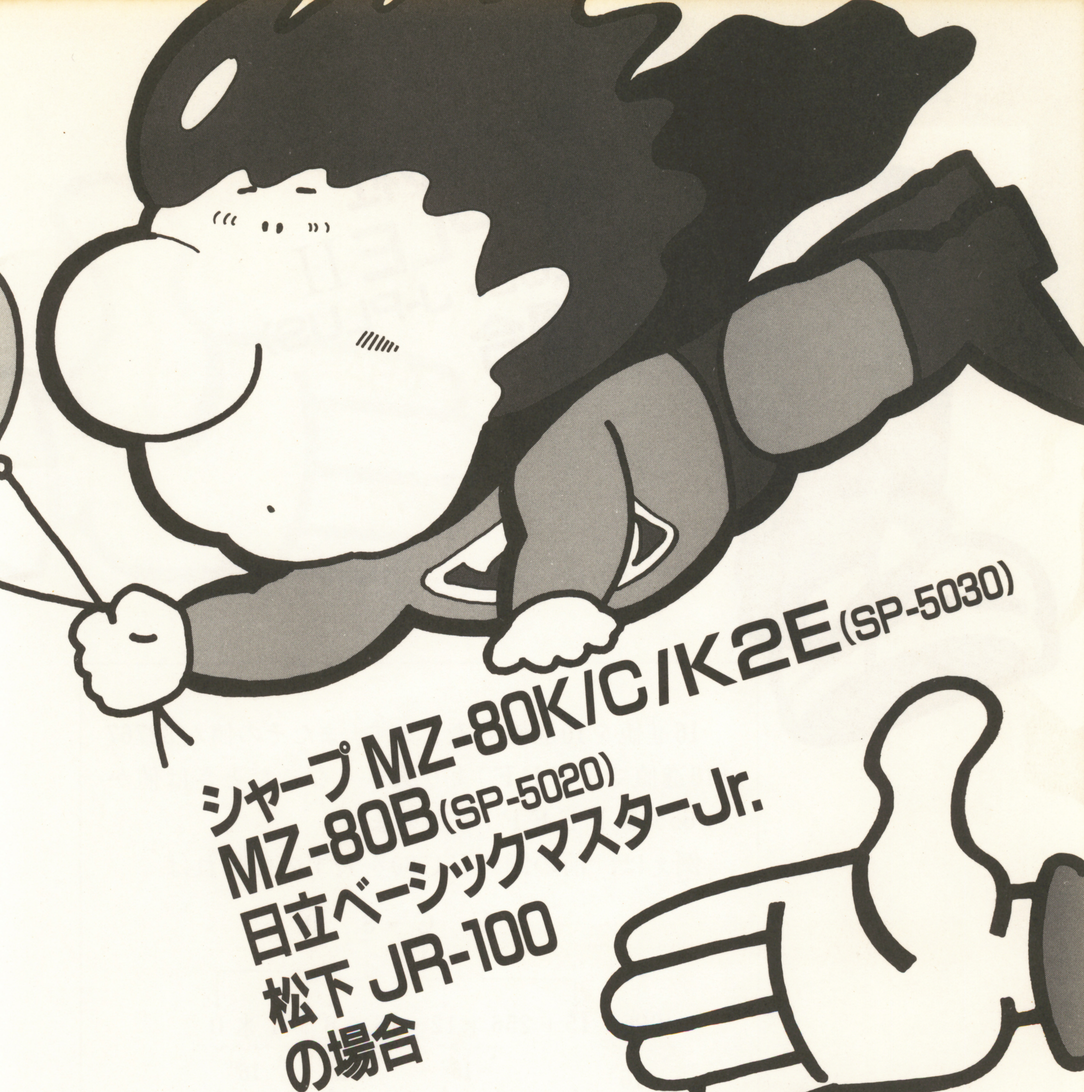




NEC PC-8001，PC-6001
東芝PASOPIA(T-BASIC)
富士通FM-8
日立ベーシックマスターレベル3
の場合

16進で表すには，16進値の頭に＆Ｈを付けます。例えば，９Ｃ00番地を表すには＆Ｈ９Ｃ00となります。

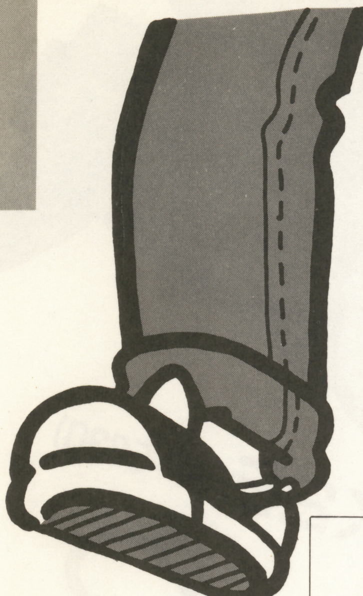




シャープMZ-80K/C/K2E(SP-5030)
MZ-80B(SP-5020)
日立ベーシックマスターJr.
松下JR-100
の場合

16進で表すには，16進値の頭に＄（ダラー・マーク）を付けます。

例えば，ＦＣ80番地を表すのであれば＄ＦＣ80となります。



アップル社 APPLE II (PLUS, J-PLUS) の場合



16進値を10進値で表します。またその値が32767（16進値で７ＦＦＦ）より大きくなったときは値から 65535(ＦＦＦＦ)を引きます。

例えば，前の例のＦＣ80を表すのであれば

$$4096 * 15 + 256 * 12 + 16 * 8 + 1 * 0$$

（Ｆ→15，Ｃ→12，8→8，0→0。4096＝16^3，256＝16^2，16＝16^1，1＝16^0，結果は10進値）

〰〰は決められた値（16進の各桁を10進値で表したときの値）で，$16^{(桁-1)}$ で表します。よって，10進値は64640となりました。結果が32767よりも大きくなったので，値から65535を引きます。

64640－65535→－895がそのアドレスになります。

＆Hや＄を使って16進を表した場合，その値を変数に代入すると，代入された変数の値はどうなるでしょうか（代入されても変数の値は16進値のまま，あるいは16進値が10進値になる。どう思いますか）。

それでは，16進値を変数に代入して表示させてみることにします。値は16進値でＦＦ，10進値では255になります。

◆リスト１－２

16進値を表わす方法"$"を使う場合もある

16進のFF，10進値で255です

```
10 A=&HFF
20 PRINT A
30 END

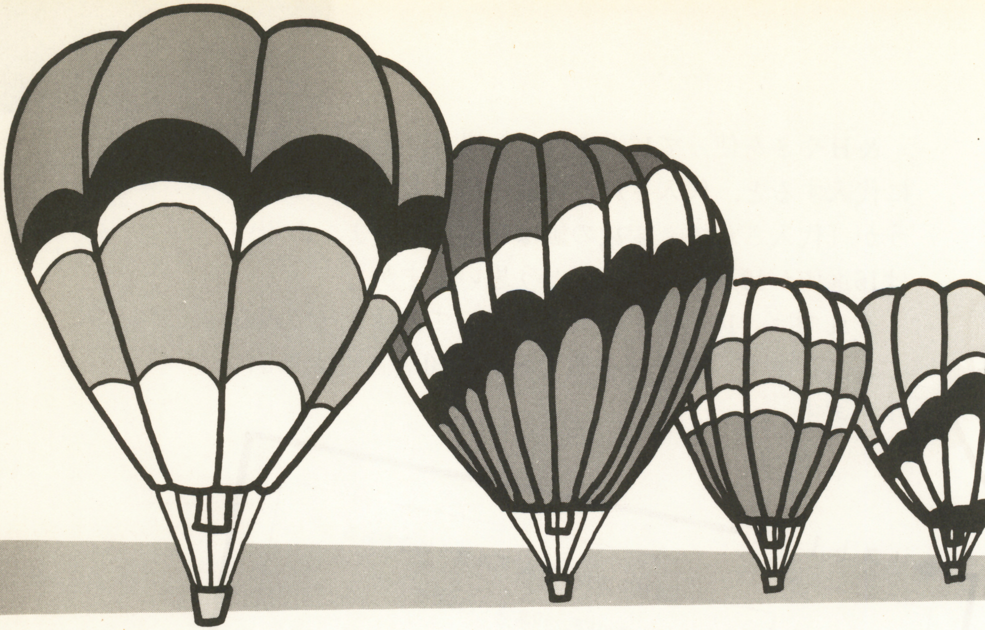
RUN
 255
```



結果は"10進値に変換される"でした。16進を10進で表したいときは，いちど変数に代入することですね。

それでは実際に使ってみることにしましょう。

㊟ＰＥＥＫ文を使ってメモリの値を読んでみると，結果は10進値となります。忘れずに!!



㋑ **あるメモリ・アドレスの内容を，ＴＶに表示させる**

アドレスは，０番地にしましょう。

プログラムは

```
10  D=PEEK(0)
             └ アドレス(メモリ番地)
20  PRINT  D
30  END
```

です。アドレスの値は０ですから，あえて16進表示することはありませんね（９以下は16進で表す必要はない)。

◆リスト１－３

```
10 D=PEEK (0)  ← アドレスの0番地の内容を見る
20 PRINT D
30 END

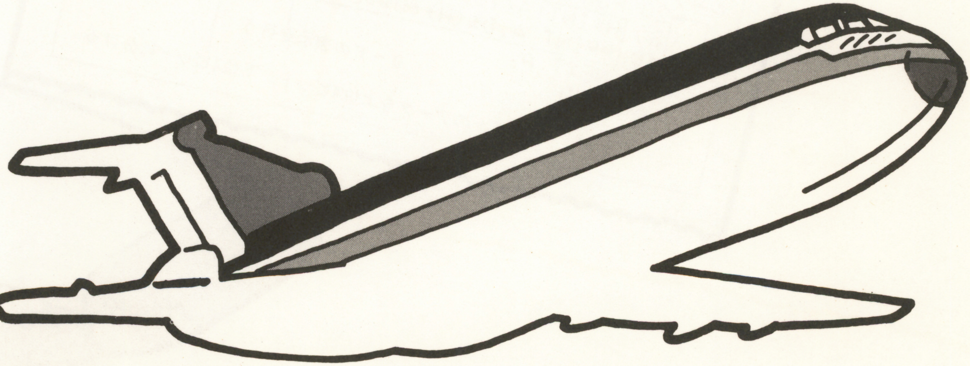
RUN
195 ← メモリの内容によって結果が異なる
```

メモリの０番地の内容が表示されました。他のアドレスでも行ってください（16進の表し方に注意してください）。

㊌あるアドレスからあるアドレスまでの内容を表示させる。

この例は，メモリ番地のあるアドレス間の内容を見るためのものです。方法はＰＥＥＫ文内のアドレスをあるアドレスからあるアドレスまで変化させればよいのです。

このとき，ＰＥＥＫ文中のアドレスの部分を，変数にしておくことがポイントです。そうすれば，ＦＯＲ～ＮＥＸＴループの変数値が利用できます。つまりＦＯＲ～ＮＥＸＴループの値は，始めのアドレスから終りのアドレスまで変化すればよいことになります。





◆リスト1－4

```
10 FOR A=&H100 TO &H105
20 PRINT PEEK(A)
30 NEXT A
40 END

RUN
195
75
13
244
43
85
```

ループ
始めのアドレス
終りのアドレス
100番地の内容
105番地の内容

㊞結果を16進で表示させる

前の例の結果を10進値ではなく，16進値で表示させようというものです。16進値で表示させるプログラムは，前に作りましたね。次のプログラムがそうです。値は 0 ～255までを16進値で表示します。

◆リスト1－5

```
10 GOSUB 100
20 FOR A=0 TO 255
30 H=INT(A/16):L=A-H*16
40 PRINT A,D$(H);D$(L)
50 NEXT A
60 END
```

0～255まで値が変化する
ループ
0～Fの表示を行う
1バイトのデータを上位4ビット，下位4ビットに分ける



◆リスト1－6

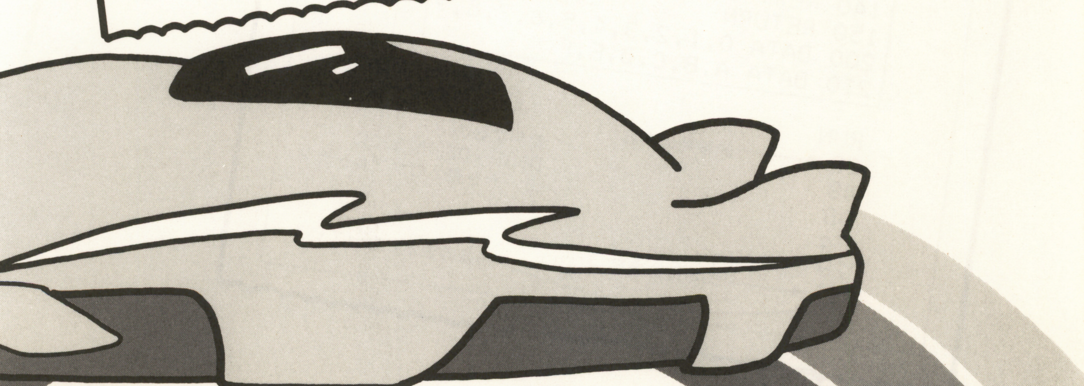
```
100 DIM D$(15)
110 FOR B=0 TO 15
120 READ F$
130 D$(B)=F$
140 NEXT B
150 RETURN
200 DATA 0,1,2,3,4,5,6,7,8,9
210 DATA A,B,C,D,E,F
```

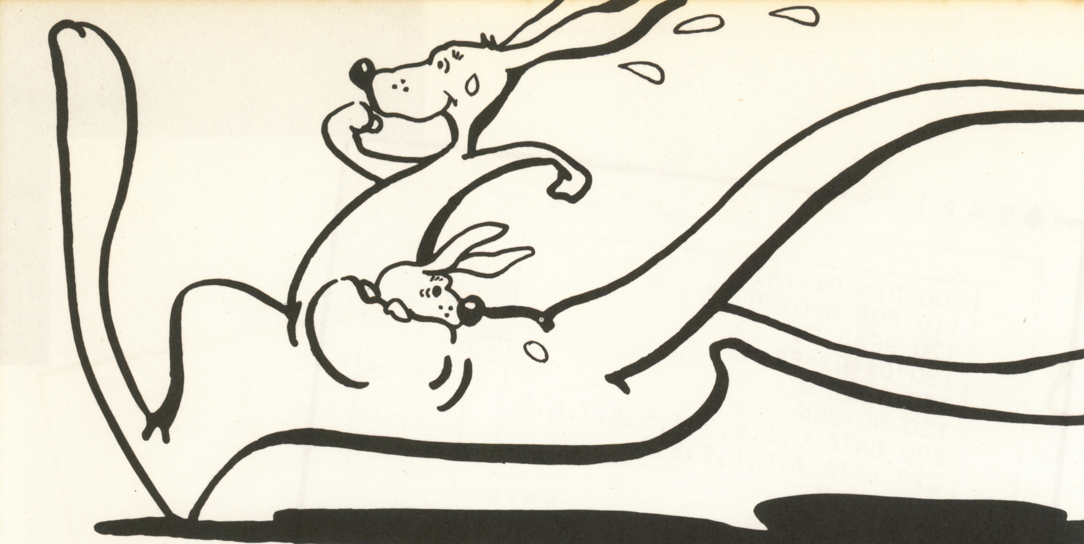
配列変数に
0～Fまでのデータ
を書込む

```
RUN
  0      00
  1      01
  2      02
  3      03
  4      04
  5      05
  6      06
  7      07
  8      08
  9      09
 10      0A
 ~~      ~~
250      FA
251      FB
252      FC
253      FD
254      FE
255      FF
```

途中省略

10進値　　16進値





これを前のプログラムとくっつけます。PEEKで読んできた値を，そのプログラムに入れればOKですね。

◆リスト1－7

```
10 GOSUB 100
20 FOR A=&H100 TO &H105
30 D=PEEK (A)
40 H=INT(D/16):L=D-H*16
50 PRINT D$(H);D$(L)
60 NEXT A
70 END
100 DIM D$(15)
110 FOR B=0 TO 15
120 READ F$
130 D$(B)=F$
140 NEXT B
150 RETURN
200 DATA 0,1,2,3,4,5,6,7,8,9
210 DATA A,B,C,D,E,F

RUN
C3
4B
0D
F4
2B
55
```

- 10 GOSUB 100 ← 配列変数にデータを書込む
- 20～60 ループ
- 40 ← 1バイトの結果を上位，下位4ビットずつに分ける。4ビットで表わすことができるのは0～F（10進値で0～15）
- 50 ← 結果を文字になおし表示
- 100～210 ← 配列変数のD$(0)～D$(15)までに0～Fを書込むサブルーチン
- C3 ← 100番地の内容
- 55 ← 105番地の内容
- C3～55 ← 16進の表示になっている

さらに応用して，結果を２進数で表示させましょう。前のプログラムのデータを，２進にするだけでできてしまいます(頭の使いようですね)。

◆リスト１－８

```
ループ 10 GOSUB 100  ←配列変数にデータを書込む
       20 FOR A=&H100 TO &H105
       30 D=PEEK (A)
       40 H=INT(D/16):L=D-H*16
       50 PRINT D$(H);" ";D$(L)
       60 NEXT A
       70 END
```

４ビットずつにデータを分け，２進を表示する

◆リスト１－９

```
100 DIM D$(15)
110 FOR B=0 TO 15
120 READ F$
130 D$(B)=F$
140 NEXT B
150 RETURN
200 DATA 0000,0001,0010,0011,0100
210 DATA 0101,0110,0111,1000,1001
220 DATA 1010,1011,1100,1101,1110,1111
```

配列変数にデータを書込むサブルーチン（100〜150）

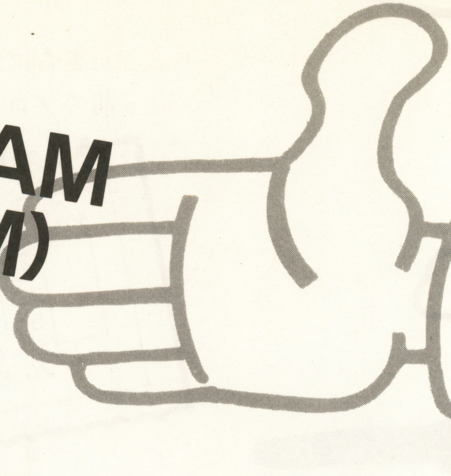
２進表示のデータ（200〜220）

```
RUN
1100 0011   ←100番地の内容
0100 1011
0000 1101
1111 0100
0010 1011
0101 0101   ←105番地の内容
```

上位４ビット　下位４ビット

８ビット→１バイトの結果

ビデオRAM (V-RAM)





パソコンには，ＴＶに画像を表示するための内容を記憶しているメモリ（ＲＡＭ）があります。このＲＡＭに直接データを書き込めば，そのデータに対応した文字が表示されます。このＲＡＭを仕事の内容からビデオ・メモリ(Ｖ－ＲＡＭ)と呼んでいます。



ビデオＲＡＭにもやはり番地（アドレス）があり，自由に読み書きすることができます。ＰＥＥＫ文を使って，Ｖ－ＲＡＭの内容を読み出してみましょう。そ

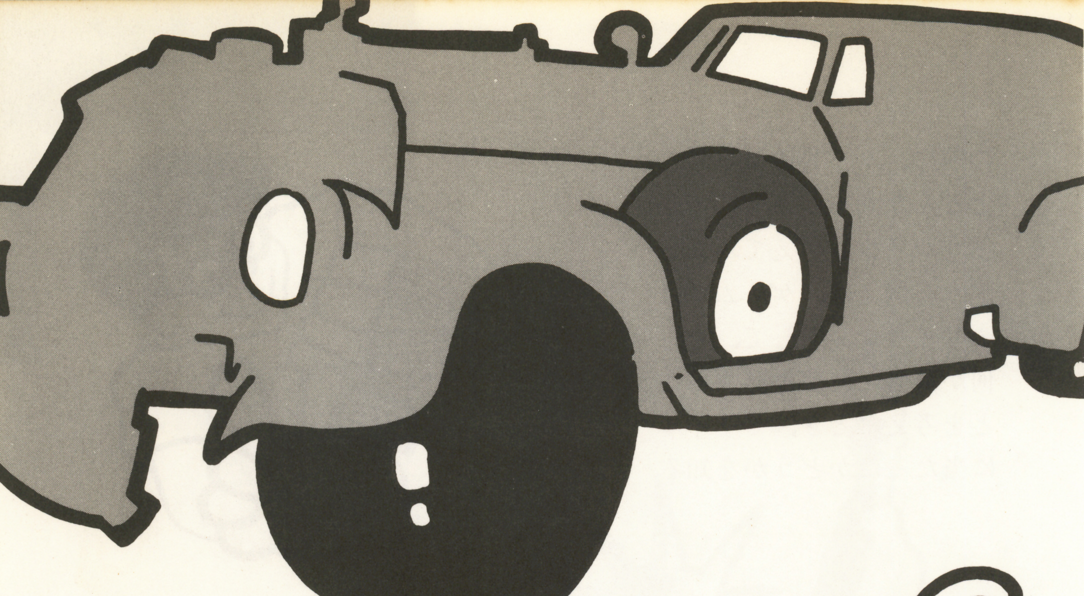
の前に，V－RAMを読むとどんな利点があるのでしょうか？少し考えてみましょう。例えば，ゲームではミサイルを発射して，もし何かに当たれば爆発することを考えたとき，ミサイルに当たったかどうかを知る





ためには，当たる物の位置があらかじめわかっている場合か，ミサイルの前に何かあるかを画面を読んで調べる方法があります（他に画面に表示されているものすべてを配列変数に入れ，物があるかないかを調べる方法もあります)。

このように，画面を読んでミサイルの前に何も表示してなければ爆発させないとすればよいのです。つまり，PEEK文でV－RAMを読むことで，どの位置に何が表示されているかを調べることができるのです。



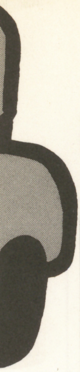
各社・パソコンの V-RAM アドレス



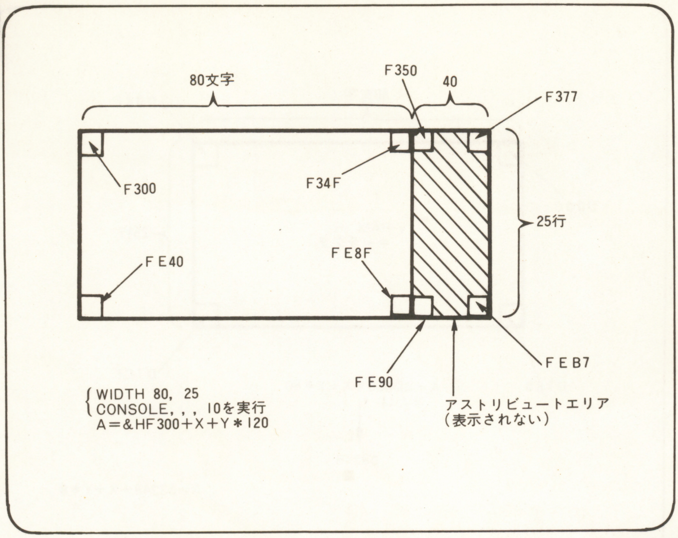
V－RAMのアドレス（住所）は，パソコンによってまちまちで，同じというのはほとんどありません。さらに最近のパソコンでは，PEEK文などを使って直接V－RAMを読むことができないマシンが多くなってきました。残念なことです。

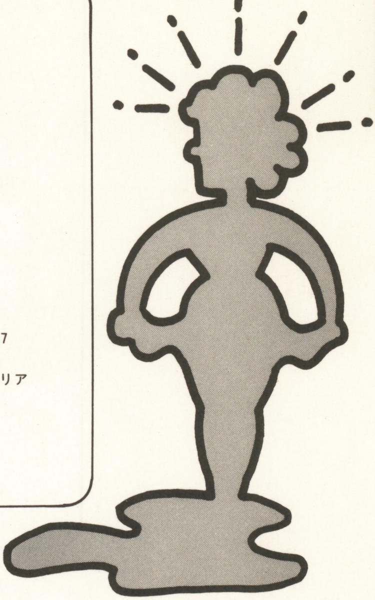
ある画面のある部分のアドレスを求めようとしたとき，カーソル・コントロール命令同様に，水平・垂直方向の二つの値から求められる方が便利なので，求めるための式も載せておきます。

アドレスを求める式に，求めたい位置の水平・垂直の二つの値を代入すれば，その位置のアドレスが出てきます。



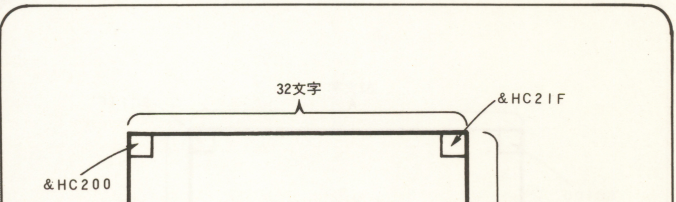
NEC PC-8001



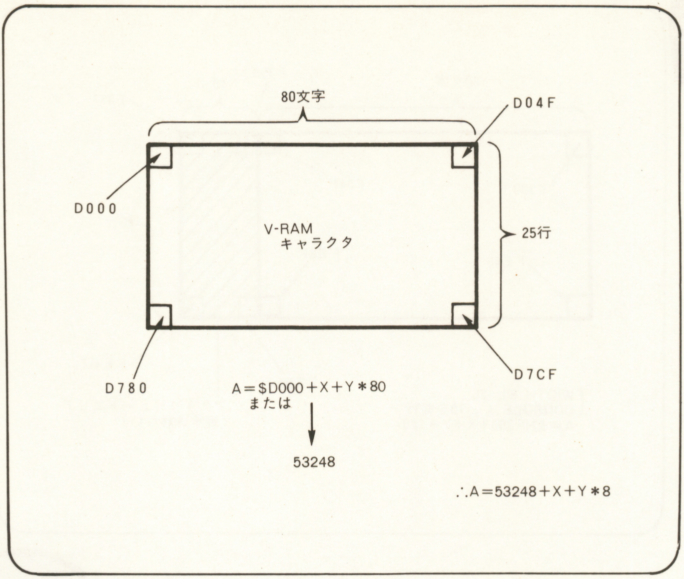




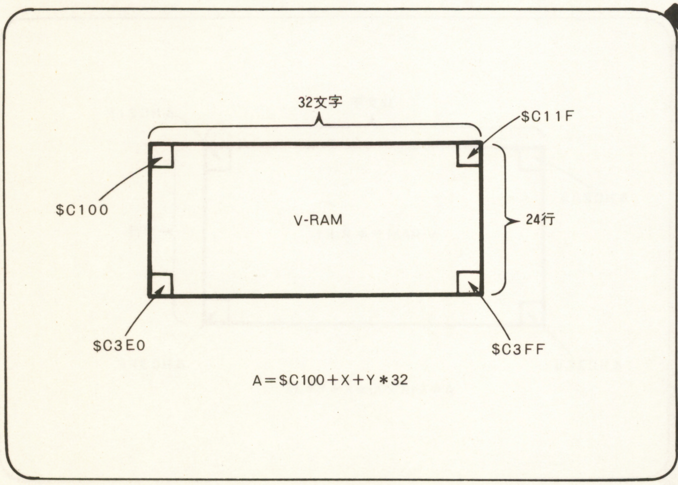
NEC PC-6001



シャープ MZ80K/C/B



松下 JR-100





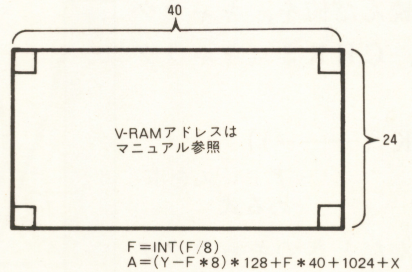
式の使い方は，カーソルのX軸方向の値が10，Y軸方向の値が5であったとすると，その位置に何が書かれているかを調べる場合，前記の式にその値を代入しアドレスの値を求め，そのアドレスをＰＥＥＫ関数で見ればよいのです。

例えばＰＣ－8001の場合では，

アドレスを求める式は，＆ＨＦ300＋Ｘ＋Ｙ＊120でしたから，この式にＸとＹの値を代入した結果をＰＥＥＫ関数に代入すればＯＫです。

```
10　X＝10
20　Y＝5
30　D＝PEEK（&HF300＋X＋Y＊120）
40　PRINT　D
50　END
```

アップル社 APPLEⅡ/PLUS/J-PLUS



V-RAMの値とキャラクタの関係

ＰＥＥＫ関数でＶ－ＲＡＭを読んだ値と，キャラクタ（文字）との関係は，各機種の説明書の中にあるキャラクタ・コード表と対応しています。機種別にキャラクタ・コード表を載せておきます。

参考 もしＰＥＥＫ関数で読んだ値を，キャラクタとして見たいのであれば，ＣＨＲ＄（キャラクタ関数）を使います。

```
A$=CHR$ (PEEK (———))
PRINT  A$
```

（———：アドレスを求める式）

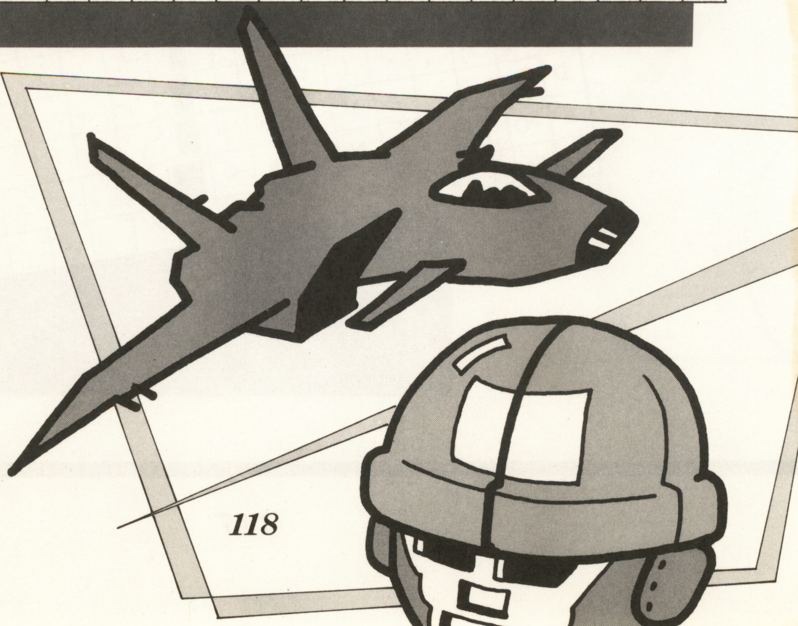
アドレスをＰＥＥＫで読んだときのキャラクタが，文字変数のＡ＄に入り，結果が表示されます。

■NEC PC-8001 キャラクタ・コード表

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		DE		0	@	P		p	▁	┴		ー	タ	ミ	═	╳
1	SH	D1	!	1	A	Q	a	q	▂	┬	。	ア	チ	ム	╞	円
2	SX	D2	"	2	B	R	b	r	▃	┤	「	イ	ツ	メ	╪	年
3	EX	D3	#	3	C	S	c	s	▄	├	」	ウ	テ	モ	╡	月
4	ET	D4	$	4	D	T	d	t	▅	▔	、	エ	ト	ヤ	◢	日
5	EQ	NK	%	5	E	U	e	u	▆	─	・	オ	ナ	ユ	◣	時
6	AK	SN	&	6	F	V	f	v	▇	│	ヲ	カ	ニ	ヨ	◥	分
7	BL	EB	'	7	G	W	g	w	█	│	ァ	キ	ヌ	ラ	◤	秒
8	BS	CN	(	8	H	X	h	x	▏	┌	ィ	ク	ネ	リ	♠	
9	HT	EM	)	9	I	Y	i	y	▎	┐	ゥ	ケ	ノ	ル	♥	
A	LF	SB	*	:	J	Z	j	z	▍	└	ェ	コ	ハ	レ	♦	
B	HM	EC	+	;	K	[	k	{	▌	┘	ォ	サ	ヒ	ロ	♣	
C	CL	→	,	<	L	¥	l	¦	▋	╭	ャ	シ	フ	ワ	●	
D	CR	←	-	=	M	]	m	}	▊	╮	ュ	ス	ヘ	ン	○	
E	SO	↑	.	>	N	^	n	~	▉	╰	ョ	セ	ホ	゛	╱	
F	SI	↓	/	?	O	_	o		┼	╯	ッ	ソ	マ	゜	╲	

■NEC PC-6001
キャラクタ・コード表

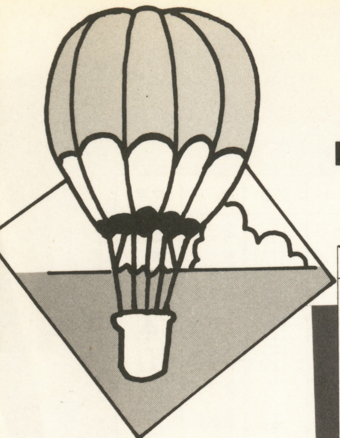
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		π		0	@	P		p	♠			ー	タ	ミ	た	み
1	月	┴	!	1	A	Q	a	q	♥	あ	。	ア	チ	ム	ち	む
2	火	┬	"	2	B	R	b	r	♣	い	「	イ	ツ	メ	つ	め
3	水	┤	#	3	C	S	c	s	◆	う	」	ウ	テ	モ	て	も
4	木	├	$	4	D	T	d	t	○	え	、	エ	ト	ヤ	と	や
5	金	┼	%	5	E	U	e	u	●	お	・	オ	ナ	ユ	な	ゆ
6	土	│	&	6	F	V	f	v	を	か	ヲ	カ	ニ	ヨ	に	よ
7	日	─	'	7	G	W	g	w	ぁ	き	ァ	キ	ヌ	ラ	ぬ	ら
8	年	┌	(	8	H	X	h	x	ぃ	く	ィ	ク	ネ	リ	ね	り
9	円	┐	)	9	I	Y	i	y	ぅ	け	ゥ	ケ	ノ	ル	の	る
A	時	└	*	:	J	Z	j	z	ぇ	こ	ェ	コ	ハ	レ	は	れ
B	分	┘	+	;	K	[	k	{	ぉ	さ	ォ	サ	ヒ	ロ	ひ	ろ
C	秒	×	,	<	L	¥	l	¦	ゃ	し	ャ	シ	フ	ワ	ふ	わ
D	百	大	-	=	M	]	m	}	ゅ	す	ュ	ス	ヘ	ン	へ	ん
E	千	中	.	>	N	^	n	~	ょ	せ	ョ	セ	ホ	゛	ほ	
F	万	小	/	?	O	_	o		っ	そ	ッ	ソ	マ	゜	ま	



■シャープ MZ80K/C/1200
キャラクタおよびコントロール・コード

コード	文字	コード	文字	コード	文字	コード	文字	コード	文字	コード	文字	コード	文字
0		37	%	74	J	111	[illegible]	148	エ	185	ル	222	▚
1		38	&	75	K	112	日	149	オ	186	レ	223	▚
2		39	'	76	L	113	月	150	カ	187	ロ	224	═
3		40	(	77	M	114	火	151	キ	188	ワ	225	♠
4		41	)	78	N	115	水	152	ク	189	ン	226	║
5		42	*	79	O	116	木	153	ケ	190	゛	227	═
6		43	+	80	P	117	金	154	コ	191	゜	228	═
7		44	,	81	Q	118	土	155	サ	192	SP	229	▔
8		45	−	82	R	119	生	156	シ	193	▐	230	▁
9		46	.	83	S	120	年	157	ス	194	▄	231	║
10		47	/	84	T	121	時	158	セ	195	▕	232	║
11		48	0	85	U	122	分	159	ソ	196	▁	233	◣
12		49	1	86	V	123	秒	160	タ	197	▏	234	╮
13	CR	50	2	87	W	124	円	161	チ	198	▒	235	╭
14		51	3	88	X	125	¥	162	ツ	199	▕	236	▁
15		52	4	89	Y	126	£	163	テ	200	█	237	╲
16		53	5	90	Z	127	⇩	164	ト	201	◤	238	╱
17	↓	54	6	91	[	128	SP	165	ナ	202	▐	239	▏
18	↑	55	7	92	\	129	°	166	ニ	203	╟	240	▕
19	→	56	8	93	]	130	「	167	ヌ	204	◙	241	●
20	←	57	9	94	↑	131	」	168	ネ	205	┘	242	▁
21	H	58	:	95	←	132	、	169	ノ	206	┐	243	♥
22	C	59	;	96	[illegible]	133	・	170	ハ	207	▁	244	▏
23		60	<	97	[illegible]	134	ヲ	171	ヒ	208	┌	245	◢
24		61	=	98	[illegible]	135	ァ	172	フ	209	╧	246	╳
25		62	>	99	[illegible]	136	ィ	173	ヘ	210	╤	247	○
26		63	?	100	[illegible]	137	ゥ	174	ホ	211	╢	248	♣
27		64	@	101	[illegible]	138	ェ	175	マ	212	▏	249	▕
28		65	A	102	[illegible]	139	ォ	176	ミ	213	▌	250	♦
29		66	B	103	☻	140	ャ	177	ム	214	▐	251	╬
30		67	C	104	☺	141	ュ	178	メ	215	▔	252	▒
31		68	D	105	[illegible]	142	ョ	179	モ	216	▀	253	║
32	SP	69	E	106	[illegible]	143	ッ	180	ヤ	217	▄	254	◥
33	!	70	F	107	[illegible]	144	ー	181	ユ	218	▕	255	π
34	"	71	G	108	[illegible]	145	ア	182	ヨ	219	╯		
35	#	72	H	109	[illegible]	146	イ	183	ラ	220	╰		
36	$	73	I	110	[illegible]	147	ウ	184	リ	221	┘		

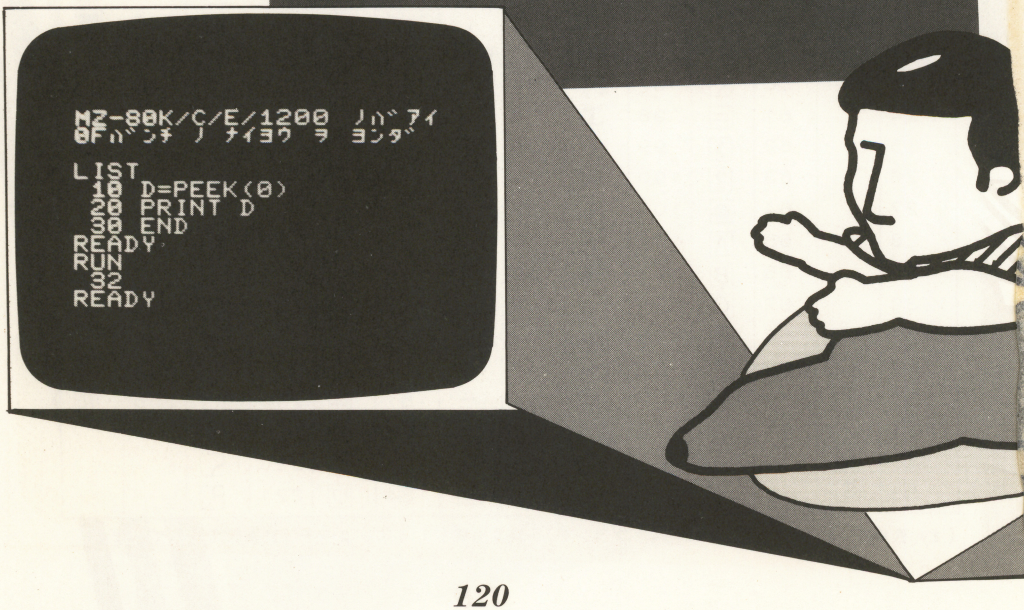
（注）表中で SP は空白（スペース，ブランク）を表すものとします。コードは10進数で表しています。



■シャープ MZ-80B キャラクタおよびコントロール・コード

上位4ビット（横）／下位4ビット（縦）

下位4ビット ＼ 上位4ビット	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NULL		（空白）	0	@	P	`	p	║	≡	V	Y	タ	ミ	Z	0
1	↓		!	1	A	Q	a	q	↓	¥	。	ア	チ	ム	A	1
2	↑		"	2	B	R	b	r	↑	U	「	イ	ツ	メ	B	2
3	→		#	3	C	S	c	s	→	●	」	ウ	テ	モ	C	3
4	←		$	4	D	T	d	t	←	○	W	エ	ト	ヤ	D	4
5	HOME		%	5	E	U	e	u	♠	┐	X	オ	ナ	ユ	E	5
6	CLR		&	6	F	V	f	v	♥	┘	ヲ	カ	ニ	ヨ	F	6
7	DEL		'	7	G	W	g	w	♦	┌	ァ	キ	ヌ	ラ	G	7
8	INST		(	8	H	X	h	x	♣	└	ィ	ク	ネ	リ	H	8
9	GRPH		)	9	I	Y	i	y	╞	┼	ゥ	ケ	ノ	ル	I	9
A	SFT LOCK		*	:	J	Z	j	z	╡	│	ェ	コ	ハ	レ	J	P
B			+	;	K	[	k	{	╥	─	ォ	サ	ヒ	ロ	K	Q
C	カナ		,	<	L	\	|	|	╨	┴	ャ	シ	フ	ワ	L	R
D			−	=	M	]	m	}	╬	┬	ュ	ス	ヘ	ン	M	S
E	L SCRIPT	■	.	>	N	^	n	~	╪	┤	ョ	セ	ホ	゛	N	T
F	カナ CANCEL	▒	/	?	O	‾	o	↴	╫	├	ッ	ソ	マ	゜	O	π

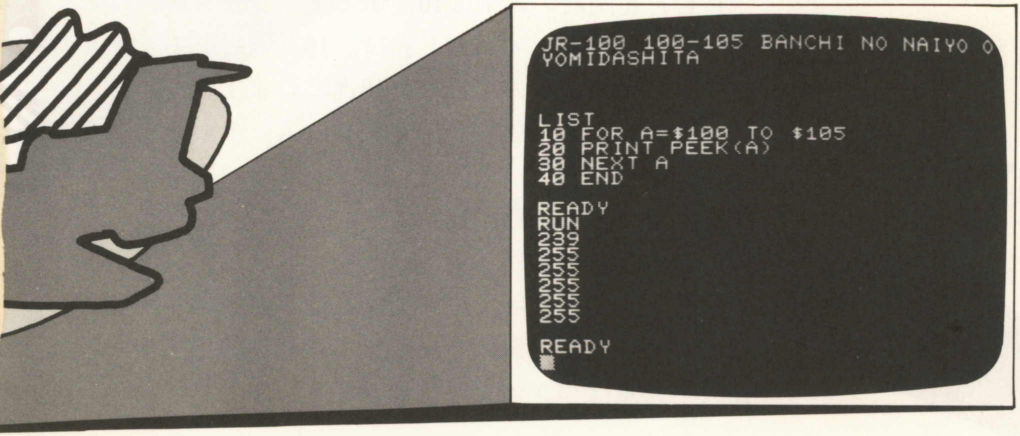


■松下 JR-100
キャラクタ・コード表

上位4ビット＼下位4ビット	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
キャラクタROM			0	1	2	3			4	5					6	7
0			(SP)	0	@	P				▟					○	⊞
1			!	1	A	Q			♠	▁					▗	□
2			▾▾	2	B	R			♥	▂					▚	◧
3			#	3	C	S			♦	▃					▛	▗
4			$	4	D	T			♣	▄					▛	◨
5			%	5	E	U			▲	▅					◧	▐
6			&	6	F	V			←	▆					▜	▜
7			▼	7	G	W			↓	▇					▚	▐
8			(	8	H	X			↑	⊞					▌	▘
9			)	9	I	Y			→	⊞					⊞	▐
A			*	:	J	Z			웃	◫					▒	▝
B			+	;	K	(			☺	⊟					⊞	◹
C			,	<	L	¥			▒	◩					⊞	●
D			—	=	M	)			◱	◲					▒	◸
E			·	>	N	^			■	⊠					▀	◿
F			/	?	O	_			◪	⧄					⊞	⧅

← ASCIIコードの場合

← ビデオRAMの場合



キャラクタ・コード表の見かた

PEEK関数で読んだ値が，何のキャラクタかを調べる



ＰＥＥＫ関数の結果は10進値で出てきますが，ほとんどのキャラクタ・コード表は，16進値対キャラクタ表になっていますので結果を16進にします。

ＰＥＥＫで読んだ値が，変数Ｄに入っているとすると，

Ｈ＝ＩＮＴ（Ｄ／16）

Ｌ＝Ｄ－Ｈ＊16

で出てきます。Ｈは上位の値，Ｌは下位の値です。そして，それぞれの値は０～15までの10進値ですから，10以上の値はつぎのように表します。

10	11	12	13	14	15	⬅10進の表示
↓	↓	↓	↓	↓	↓	
A	B	C	D	E	F	⬅16進の表示

（注）0～9まではそのまま



㊉ Dの値が65であったときのキャラクタ・コードは？

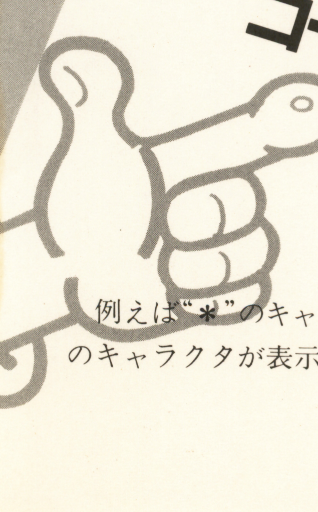
H＝INT（65／16）

L＝65－H＊16

よって　H→4，L→1　キャラクタ・コードは41になります。そこで，表の上位4のところと下位1のところを見ると，“A”と出てきます。

参考　なにも表示していない画面は，空白を表示しているわけですから，10進値で32，16進値で20が出てきます。キャラクタ・コード表で確かめてください。

キャラクタからコードを求める



例えば“＊”のキャラクタ・コードを求める場合，“＊”のキャラクタが表示されている位置からX，Y方向に

線を延ばし，当たったところのコードを読みます。この場合だと，上位が２，下位がＡとなりますから，コードは２Ａとなるわけです。

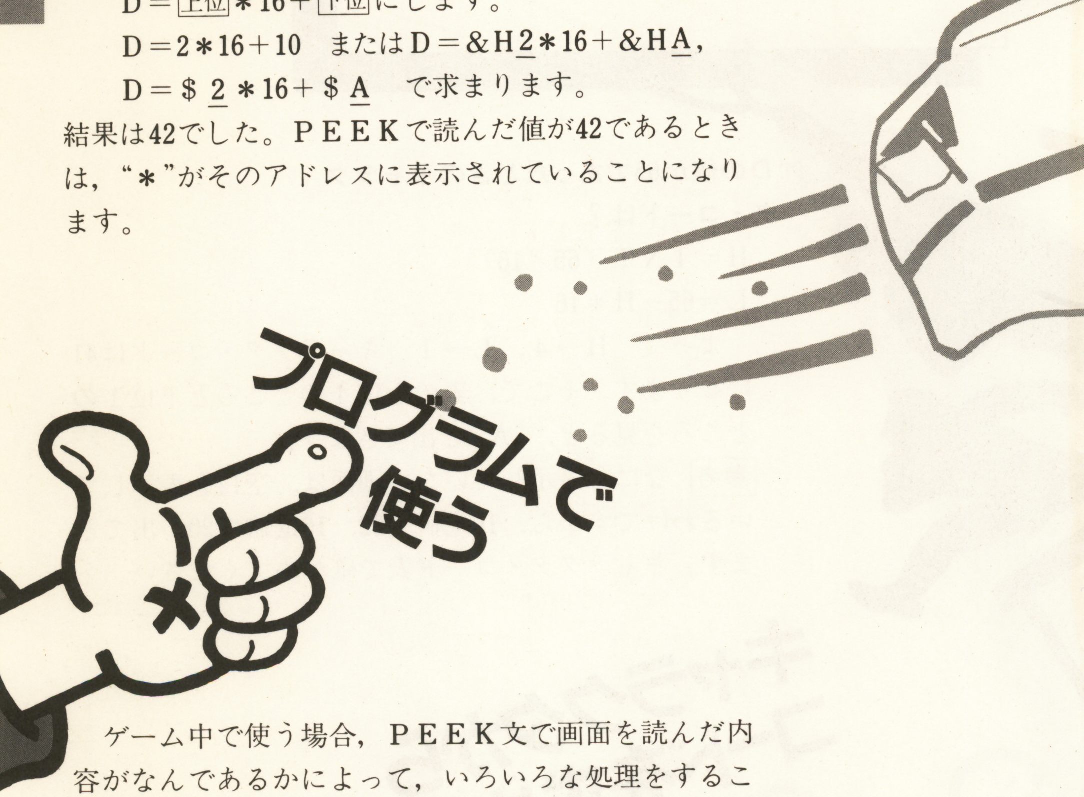
10進値にする場合

D＝上位＊16＋下位にします。

D＝2＊16＋10　またはD＝＆H2＊16＋＆HA，

D＝＄２＊16＋＄A　で求まります。

結果は42でした。ＰＥＥＫで読んだ値が42であるときは，“＊”がそのアドレスに表示されていることになります。



プログラムで使う

ゲーム中で使う場合，ＰＥＥＫ文で画面を読んだ内容がなんであるかによって，いろいろな処理をすることができます。これは，ＩＦ文を使って処理します。

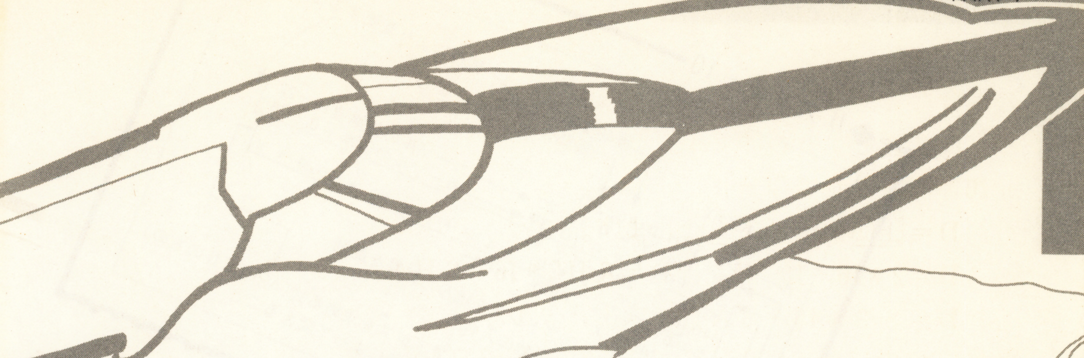
例えば，ある位置に“＊”が表示されているかを調べるには，

Ｄ＄＝ＣＨＲ＄（ＰＥＥＫ（――））

↑ ある位置のアドレスを求める式

ＩＦ　Ｄ＄＝“＊”　ＴＨＥＮ “＊”のときの処理

とすれば，“＊”が表示されているときに，別の処理を



させることができます。

CHR＄関数を使わないで，同じことをするのなら

D＝PEEK（――）
↑
アドレスを求める式

IF　D＝42　THEN　［“＊”のときの処理］
↑
“＊”の10進値または＆H2A，＄2A（16進値）

となります。このように，IF文によって読んだ値がなんであるかをチェックし，処理をします。

㋑画面表示すべてを配列変数に入れる

V－RAMで読んだ値を，配列変数に入れて記憶します。そのため画面の表示が変になっても，再び配列変数の値をキャラクタに変更して表示すれば，元の画面に戻すことができます（PC－8001やFM－8などの中にある，GET＠やPUT＠Aなどの命令と，同じようなことができる）。

配列変数に記憶した内容を表示する。前のプログラムを実行したあと，このプログラムを実行すると，記憶した画面を表示します。㊟ただし，実行するときはGOTO100として実行してください。RUN100とすると，配列変数の内容が消えます。しかし前のプログラムがENDなどで止まらず，プログラム実行中であれば，そのままこのプログラムを実行できます。

◆リスト1－10

```
JLIST
10  DIM D(40,25)
20  FOR Y = 0 TO 22
30  FOR X = 0 TO 39
35  F = INT (Y / 8)
40  A = (Y - F * 8) * 128 + F * 40 + 1024 + X
50  D(X,Y) = PEEK (A)
60  NEXT X
70  NEXT Y
80  END
```

横40文字
縦25行
機種により変更すること
カーソルの値垂直10～22
カーソルの値水平0～39
ループ
アドレスを読んだ結果を配列変数に入れる
アドレスを求める式(APPLEⅡのもの)
各機種によって変えること



◆リスト1－11

```
JLIST
100  FOR Y = 0 TO 22
110  FOR X = 0 TO 39
115  HTAB X + 1: VTAB Y + 1
120  PRINT CHR$ (D(X,Y))
130  NEXT X
140  NEXT Y
150  END
```

垂直方向の値
水平方向の値
機種により上限を変えること
カーソル命令
APPLEⅡ以外は変更すること
PC-8001, PC-6001ではLOCATE X, Y
MZ80K/BではCURSOR X, Y
JR-100ではCURSOR Y, X

◆リスト1－12

パソコンが出すプロンプト(入力要求マーク)ですから
自分で入れる必要はない。出ないマシンの方が多い

4096(10進)番地

```
]PRINT PEEK(4096)
76
]POKE 4096,0
]PRINT PEEK(4096)
0
```

書込む前のメモリの内容(これ以外の値が出るかもしれない)

0を書込む

前の値76が0に書替わっている



◆リスト1－13

スタートアドレス

エンドアドレス

もちろん16進で表わしてもよい

```
]LIST
10  FOR A = 4096 TO 4100
20  READ F
30  POKE A,F
40  NEXT A
50  END
100 DATA 1,2,3,4,5
```

読んだ値をアドレスに書込む

読まれる

ループ

◆リスト 1－14

```
]POKE 4096,ASC("A")
]PRINT PEEK(4096)
65
]PRINT CHR$(PEEK(4096))
A
```

- 書込むアドレス
- 文字を" "で囲むか文字変数を使う
- アスキーコードに直す
- 書込んだ内容を見てみる
- "A"のアスキーコード
- アドレスの内容
- キャラクタに直す
- 文字(キャラクタ)になって出た

◆リスト 1－15

```
]LIST
  10
  30  INPUT A$
  40 F = 1
  50  POKE 4096, LEN (A$)
  60  FOR A = 4097 TO 4097 + LEN (A$) - 1
  70  POKE A, ASC ( MID$ (A$,F,1))
  80 F = F + 1
  90  NEXT A
      END
```

- ループ
- 文字列を入力する
- 文字の長さを書込む
- 書込むスタートアドレス
- 1文字ずつ取出して
- アスキーコードに直す
- 文字をアスキーコードに直し、アドレスAのメモリに書込む
- 文字を取出す位置を1つ増す





◆リスト1−16

```
]LIST
100 F = PEEK (4096)
110  FOR A = 4097 TO 4097 + F - 1
120  PRINT  CHR$ ( PEEK (A));
130  NEXT A
140  END
]RUN
SAICHI-TANJI
```

ループ

文字列の長さを書込んだアドレス

書込んだスタートアドレス

メモリを読む

キャラクタを直す

セミコロン";"で表示をくっつける

データの長さ−1のアドレスまで実行

書込んだ内容が出てきた

◆リスト1−17

```
]LIST
10    INPUT A
20  H =  INT (A / 256)
30  L = A - H * 256
40    POKE 4096,L
50    POKE 4097,H
60    END
]RUN
?65535
]RUN
?65536
?ILLEGAL QUANTITY ERROR IN 50
```

値を入力

1バイトずつのデータに分ける⇒上位1バイト，下位1バイトになる

下位を書込む

上位を書込む

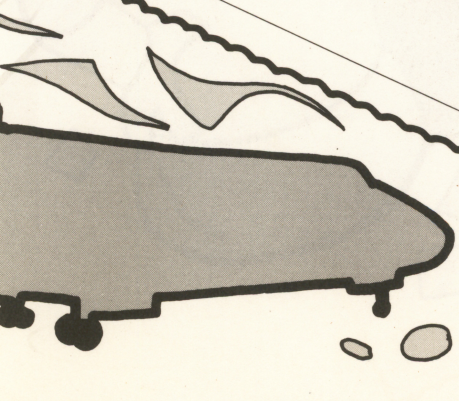
2バイトのアドレスに書込む

値を入れる

値は0〜65535までです。それ以上では，さらにメモリが必要







◆リスト1－18

```
]LIST

100 A =  PEEK (4096) + PEEK (4097) * 256
110  PRINT A
120  END

]RUN
65535
```

下位を書込んだアドレス

上位を書き込んだアドレス

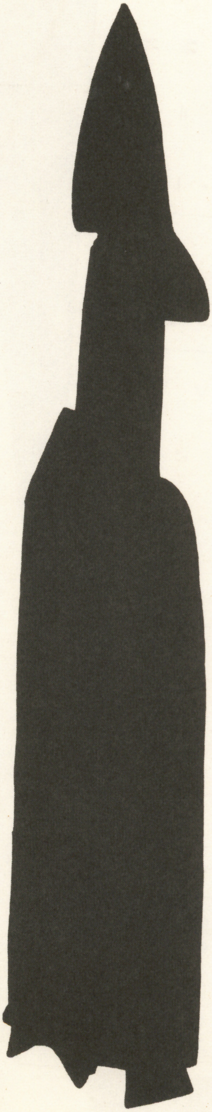
上位の値に256を掛ける

書込んだ内容が出てきた





PEEK文のまとめ



パーソナル・コンピュータに使われているメモリには，2種類ありました。RAMとROMがそれです。RAMは読み書きのできるメモリで，ROMは読みだしだけのメモリです。

これらメモリには，アドレス（番地）があって，ここにいろいろなデータが記憶されています。

PEEK文は，メモリのあるアドレスの内容を読み出すための命令です。その使い方は，次のようになります。

メモリの内容が代入される

↙

［変数］＝PEEK（［アドレス］）

↑ “ピークと読む”

↑　↑ 必ずカッコを付ける

アドレスは，10進値で表す機種と16進値で表す機種の2種類がありました。表しかたの違いを，表にします（**第1－17表**）。

■アップル社 APPLEⅡ/PLUS/J-PLUS

		Inverse				Flashing				Control		Normal				Lower case	
Desimal		0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
	16進	$00	$10	$20	$30	$40	$50	$60	$70	$80	$90	$A0	$B0	$C0	$D0	$E0	$F0
0	$0	ⓐ	P		0	ⓐ	P		0	ⓐ	P		0	ⓐ	P		0
1	$1	A	Q	!	1	A	Q	!	1	A	Q	!	1	A	Q	!	1
2	$2	B	R	"	2	B	R	"	2	B	R	"	2	B	R	"	2
3	$3	C	S	#	3	C	S	#	3	C	S	#	3	C	S	#	3
4	$4	D	T	$	4	D	T	$	4	D	T	$	4	D	T	$	4
5	$5	E	U	%	5	E	U	%	5	E	U	%	5	E	U	%	5
6	$6	F	V	&	6	F	V	&	6	F	V	&	6	F	V	&	6
7	$7	G	W	'	7	G	W	'	7	G	W	'	7	G	W	`	7
8	$8	H	X	(	8	H	X	(	8	H	X	(	8	H	X	(	8
9	$9	I	Y	)	9	I	Y	)	9	I	Y	)	9	I	Y	)	9
10	$A	J	Z	*	:	J	Z	*	:	J	Z	*	:	J	Z	*	:
11	$B	K	[	+	;	K	[	+	;	K	[	×	;	K	[	+	;
12	$C	L	/	,	<	L	/	,	<	L	/	,	<	L	/	,	<
13	$D	M	]	–	=	M	]	–	=	M	]	–	=	M	]	–	=
14	$E	N	^	.	>	N	^	.	>	N	^	.	>	N	^	.	>
15	$F	O	-	/	?	O	–	/	?	O	–	/	?	O	–	/	?



メモリを直接アクセスする命令 POKE

さて，それでは続いて「ＰＯＫＥ文」について説明します。



POKE文について

ＰＥＥＫ文はメモリの内容を見る命令でしたがその逆で，データをメモリに書き込む命令を見てみましょう。命令は“ＰＯＫＥ”文で，ポークと読みます。

10進値または16進値

POKE 書き込むアドレス , 書き込むデータ

↑ ポーク文

↑ 10進値で0～255までの値

アドレスの表しかたは，PEEKで使った方法を使います。

この命令は，データをアドレスに書き込みます。一つのアドレスは8ビット・1バイトですから，10進値で0～255までの値しか書き込めません。

㊖1000番地に99を書き込む

```
POKE  &H1000, 99
```

&H1000 → $1000や4096

99 → 16進で表すのなら &H63または$63

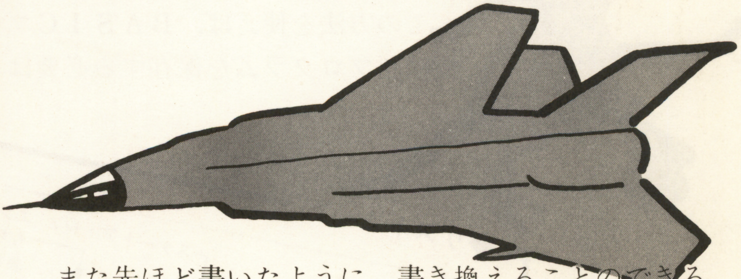


POKE文を使うときの注意





POKE文は，メモリの内容（RAMの内容）を自由に変えることができます。それだけに，間違って値を書き換えてしまうと，プログラムがこわれたり，パソコンが暴走したりしますから，書き込むアドレスには十分な注意が必要です。まずは取扱説明書の中にある“メモリ・マップ”を見て，自分が自由に使うことのできるメモリ・アドレス領域を使用してください。



また先ほど書いたように，書き換えることのできるメモリは，RAMの入っているアドレスだけで，ROMの入っているアドレスは書き込んでも書き換えることはできません。

㊀ メモリにデータを書き込んでみる

書き込む前のアドレスの内容と，書き込んだ後の内容を見る。

◆リスト1－19

```
]PRINT PEEK(4096)
76
]POKE 4096,0
]PRINT PEEK(4096)
0
```

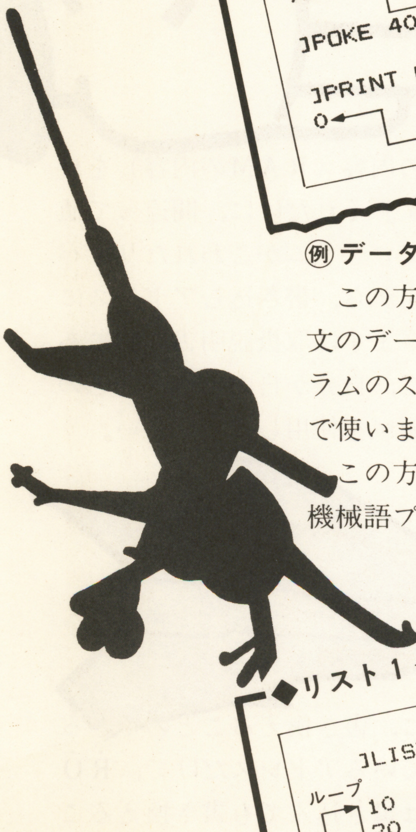
パソコンが出すプロンプト(入力要求マーク)ですから自分で入れる必要はない。出ないマシンの方が多い

4096(10進)番地

書込む前のメモリの内容(これ以外の値が出るかもしれない)

0を書込む

前の値76が0に書替っている



㊀データ文で定義したデータをメモリに書き込む

この方法は，あらかじめ機械語プログラムをデータ文のデータに定義しておき，それをＢＡＳＩＣプログラムのスタート時に，メモリに書き込むといった方法で使います。

この方法を使えば，ＢＡＳＩＣプログラムの他に，機械語プログラムを保存する必要はなくなります。

◆リスト1－20

```
]LIST
10  FOR A = 4096 TO 4100
20  READ F
30  POKE A,F
40  NEXT A
50  END
100  DATA  1,2,3,4,5
```

スタートアドレス　エンドアドレス

もちろん16進で表わしてもよい

ループ

読んだ値をアドレスに書込む

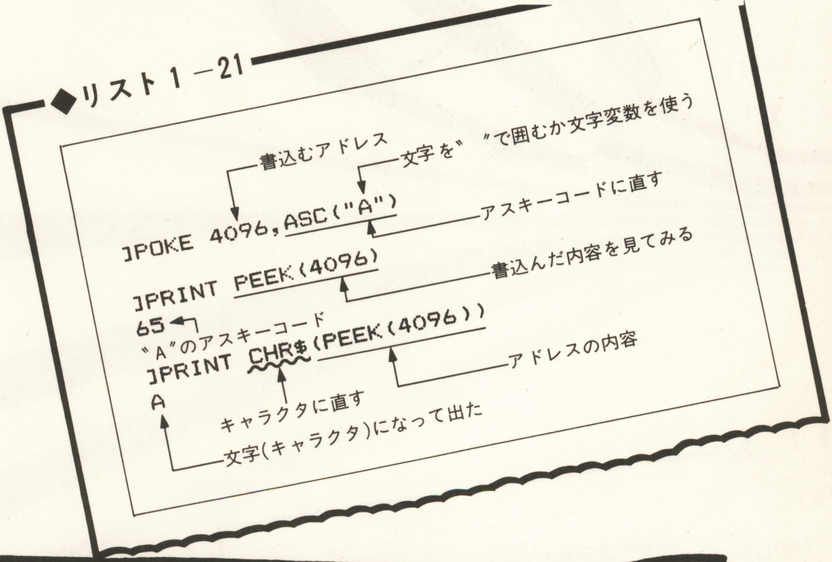
読まれる

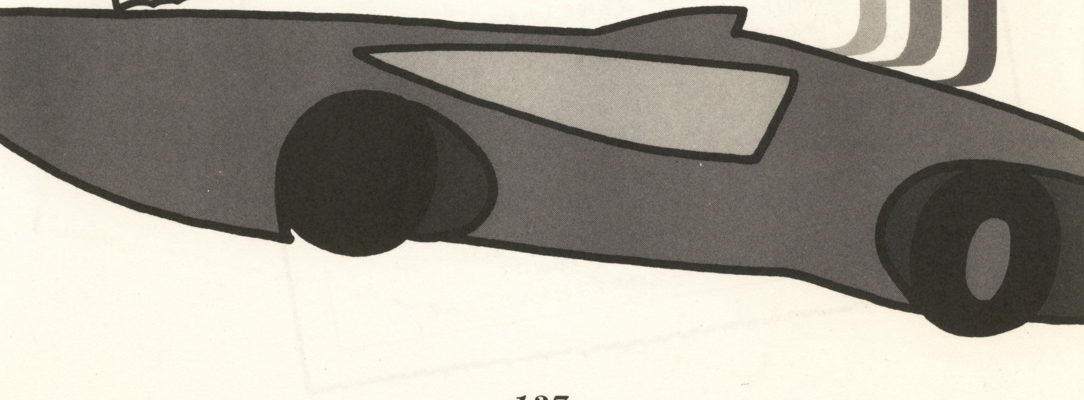
㊝キャラクタをメモリに書き込む

メモリに書き込むことができるのは数値のみですから，キャラクタ（文字）をメモリに書き込むためにはキャラクタをキャラクタ・コードなどの数値に直し，その数値を書き込むといったことが必要です。そのためには，文字をアスキー・コード（キャラクタ・コードとほぼ同じ）に直すため，ＡＳＣ（アスキー）関数というものを使います。

この関数の逆は，ＣＨＲ＄(キャラクタ)関数です。

◆リスト１－21





㊑キー・ボードから入力した文字列をメモリに書き込む

前の例を応用して，メモリに書き込みます。ポイントは，入力した文字列の長さを調べ，その分だけ１文字ずつメモリに書き込むことです。また書き込んだ文字の数がわからなくては，再びメモリを読んで，文字に直すことはできませんから，文字列の長さもメモリに書き込んでおきます。



◆リスト１－22

```
]LIST

10   INPUT A$
30   POKE 4096, LEN (A$)
40 F = 1
50   FOR A = 4097 TO 4097 + LEN (A$) - 1
60   POKE A, ASC ( MID$ (A$,F,1))
70 F = F + 1
80   NEXT A
90   END

]RUN
?SAICHI-TANJI
```

- 10：文字列を入力する
- 30：文字の長さを書込む
- 50：書込むスタートアドレス
- 50～80：ループ
- 60：ASC：アスキーコードに直す
- 60：MID$ (A$,F,1)：１文字ずつ取出して
- 60：文字をアスキーコードに直し，アドレスAのメモリに書込む
- 70：文字を取出す位置を１つ増す
- ?SAICHI-TANJI：名前の文字列を入力してみる

メモリを読んでキャラクタで表示するプログラムは次のとおりです。CHR$関数を使ってキャラクタに直します。



◆リスト1―23

```
]LIST
100 F = PEEK (4096)
110   FOR A = 4097 TO 4097 + F - 1
120   PRINT  CHR$ ( PEEK (A));
130   NEXT A
140   END

]RUN
SAICHI-TANJI
```

255以上の数をメモリに書き込む



８ビット・１バイトでは，0～255までの値しか表すことができません。そこで２バイトを使い，より大きな値を書き込んでみようというものです。

２バイトが使えると，256＊256⇒65536。よって0～65535までの値を表すことができます。

ＰＯＫＥ文では，１バイト255までの値しか書き込めませんから，工夫が必要です。

```
H＝ＩＮＴ（値／256）
L＝値－H＊256
```

とします。値を入れ計算すれば，２バイトの値HとLが出てきます。これをアドレスを変えて，一つずつ書き込めばよいのです。

◆リスト1－24

```
]LIST

10  INPUT A
20  H =  INT (A / 256)
30  L = A - H * 256
40  POKE 4096,L
50  POKE 4097,H
60  END

]RUN
?65535

]RUN
?65536

?ILLEGAL QUANTITY ERROR IN 50
```

値を入力
1バイトずつのデータに分ける
上位1バイト，下位1バイトになる
下位を書込む
2バイトのアドレスに書込む
上位を書込む
値を入れる
値は0～65535までです。それ以上では，さらにメモリが必要

書き込んだ値を元に戻すには，次のようにします。

◆リスト1－25

```
]LIST

100 A =  PEEK (4096) + PEEK (4097) * 256
110  PRINT A
120  END

]RUN
65535
```

下位を書込んだアドレス
上位を書込んだアドレス
上位の値に256を掛ける
書込んだ内容が出てきた

以上でＰＯＫＥ文の説明は終わりです。





最後に



「メモリを直接アクセスする命令，PEEKとPOKE」は，まったく逆の関係にあるわけです。

メモリの内容を読む		メモリに数値を書き込む
PEEK	←――――→	POKE

この二つの命令語はメモリ間のデータのやり取りを可能にし，機械語をベーシックで扱う上で特に重要です。ぜひマスターしてください。

編集長：分ったかな？

編，影：よく分りました。

影：編集長！よく分りました？

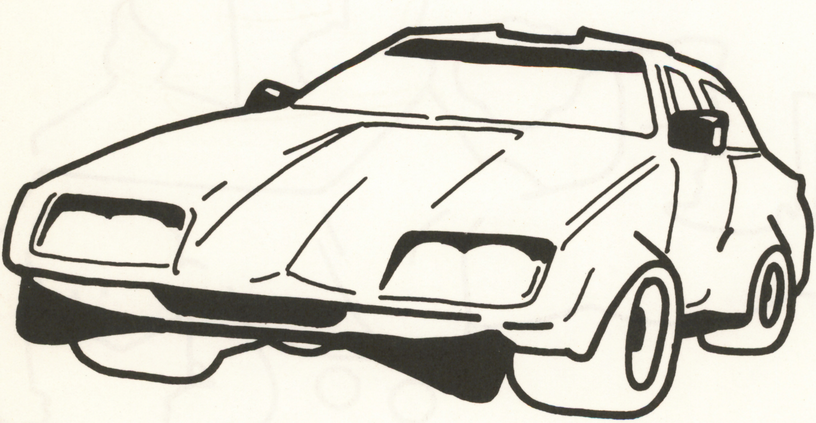
編集長：よく分った。丹治先生，またよろしくお願いします。

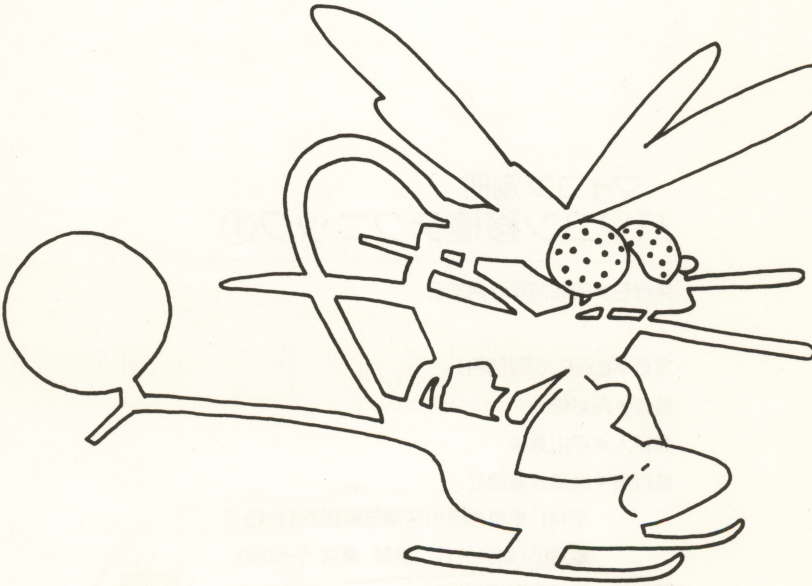
丹治先生：ハイ！…それでは帰ります（ガチャ）。



■第1－17表　アドレスの表わし方

機　　種	アドレスの表わし方
NEC　PC－8001 PC－6001	16進で表わし，アドレスの頭に＆Hを付ける
東　芝　PASOPIA （T－BASIC）	〃
富士通　FM－8	〃
日　立　ベーシックマスター レベル3	〃
シャープ　MZ80K/C/K2E MZ80B	16進で表わし，アドレスの頭に＄を付ける
日　立　ベーシックマスター Jr	〃
松　下　JR－100	〃
アップル社　APPLEII	10進値でアドレスを示す





マイコン別冊
パソコン移植テクニック①

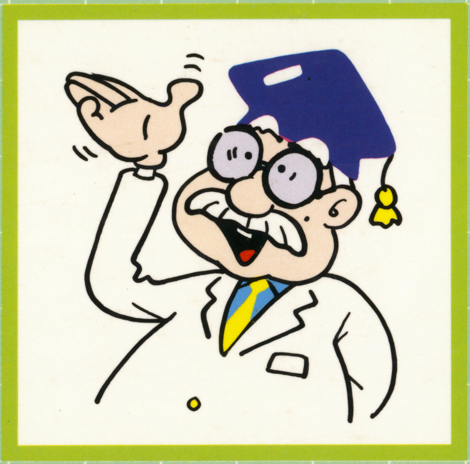
発行日●昭和59年8月30日

定価●800円（〒250円）
監修●丹治佐一
発行人●平山秀雄
発行所●㈱電波新聞社
〒141 東京都品川区東五反田1-11-15
電話03(445)6111 振替 東京 5-51961
印刷所●奥村印刷㈱
製本所●㈱堅省堂

定価800円

続編の
「パソコン移植テクニック②・UFOゲーム
大作戦」
「パソコン移植テクニック③・BASIC言
語辞典」
もよろしく!!

マイコン別冊　パソコン移植テクニック①　昭和五十九年八月三十日発行



電波新聞社　〒141 東京都品川区東五反田1－11－15　電話（03）445－6111（大代）　定価800円　雑誌68469-12

PART 1 入門編 BASIC大研究

パソコン移植テクニック 1 BASIC大研究

電波新聞社

PART 1
入門編
BASIC
大研究

パソコン移植テクニック① BASIC大研究

電波新聞社